سطر الأوامر نظرة عن قرب أكثر

بسم الله الرحمن الرحيم

بعون من الله وفضله سوف أقوم بتقديم نظرة تكاد تقترب من الشمولية لموضوع مهم وهو سطر الأوامر والذى يعتبر خاصية من أهم خصائص لينوكس على الإطلاق . ولكى تلمس ذلك على سبيل المثال لو قمت بإدارة أحد السيرفرات حينئذ الخيار الأمثل أمامك هو سطر الأوامر لقوته ومرونته وسروعته فى تنفيذ الأوامر بغض النظر عن المميزات الأخرى وتتمثل فى الأمان فى قلة عدد الثغرات المتاحة أمام المخربين.

وإن شاء الله الموضوع مقسم إلى قسمين الأول للموضوع ذاته وسيكون هنا والآخر لوضع الردود والاستفسارات على هذا الرابط:

http://www.linuxac.org/forum/showthread.php?t=1603

حتى يكون الموضوع مسلسل ويخرج إن شاء الله بصورة مشرفة .

فى البداية حينما يتبادر إلى الأذهان سطر الاوامر تجد أفكار غريبة لدى الأشخاص بخصوص هذا الموضوع منها عمليات معقدة وأوامر صعبة لا يعرف كيفية استخدامها إلا المحترفين وإلخ من هذه الأفكار .قد يكون الموضوع من جهة صحيح ولكن بالفعل وعن تجربتى الشخصية لاستخدام اللينوكس لا يمكن الفرار من سطر الأوامر سواء كنت مبتدىء أو متوسط المستوى أو محترف .فمثلا سطر الأومر يُمكّنكَ بمجرد كتابة أمر واحد أن تنفذ ما لا يمكن تنفيذه من خلال الواجهات الرسومية مهما كانت قوة الواجهة الرسومية المستخدمة سواء كانت Gnome او KDE أو أين كانت الواجهة.

كانت هذه مقدمة بسيطة عن ماهية سطر الأوامر وما يمكن أن يقوم به من مهام ولكن ما ذكرناه كانت نبذة بسيطة . سطر الأوامر بحر كبير من المعرفة يزداد فيه زادك بالعمل اليومى عليه ومعرفه مهمة ووظيفة كل أمر .

يوجد بعض الأنواع من الطرفية أو الشل أو سطر الأوامر مسميات لمعنى واحد الواجهة التى تقوم أنت بكتابة ما يحلو لك من اوامر فيها مثل طرفية csh , tcsh , zsh ولكن الطرفية التى سوف نعمل عليها من خلال الشرح بإذن الله هى طرفية bash والتى ترمز إلى Bourne again shell وسوف نتناول شرح مفصل وبالتدريج لأوامر كثيرة ومفيدة بإذن الله .

الجزء الأول : أساسيات لابد من الإلمام بها

فى هذا الجزء إن شاء الله سوف نقوم باستعراض بعض الأساسيات المهمة والتى ينبغى لمستخدم اللينوكس اليومى معرفتها والإلمام بها ونقسمها فى عناصر تالية :

1- كل شىء في عالم اللينوكس يعتبر ملف (لا تقلق سوف أقوم بشرح كل عنصر على حده)
2- أقصى طول لأسماء الملفات على اللينوكس
3-الأسماء على لينوكس حساسة تجاه الحروف الكبيرة والصغيرة
4- بعض العلامات الخاصة تتجنب كتابتها فى أسماء الملفات
5- ال WildCards وما ظيفتها ؟

طبعا من القراءة الأولية للعناصر لا يستطيع الفرد العادى من معرفة المراد منها ولذلك نبدأ أولا مع أول

عنصر :

- كل شىء فى عالم اللينوكس ملف -

فى هذا البند غموض ما بمعنى البعض يتعامل بشكل يومى مع أنظمة التشغيل المختلفة سواء كانت أنظمة لينوكس أو ماك أو حتى ويندوز ولا يعلم غير شىء وحيد وهو أن ما يسمى ملف هى الملفات التى يمكن تحريرها سواء كانت بالكتابة ، آخرون قد يعلمون أن الصوتيات والفيديوهات تعتبر ملفات ، آخرون يعلمون أن الصور ملفات وهكذا .... وهذا صحيح ولكن ما أود أن أصل إليه أنهم يعتقدون أن هذه هى الملفات فقط ولا شىء سواها على سبيل المثال القرص الصلب من معناه الظاهرى كتلة من تركيبة معادن لا يمكن التعامل معها إلا من خلال أسلاك ال Data وال power ولا يمكن التغيير فى بيانات الهارد من خلال استخدام نظام التشغيل!

جاء لينوكس ليغير كل هذه المفاهيم ويعتبر أى شىء يخص الحاسب ملف سواء كان قرص صلب ، كروت الشبكات ، كروت الشاشة أى عتاد بجانب البرامج يعتبر ملف أيضا .

البعض قد يسغرب ذلك ولكن كيف للينوكس أن يقوم بذلك ؟

الإجابة على هذا السؤال بسيطة جدا وهى أن أى شىء يقوم لينوكس بالتعامل معه يعتبره مجرد سيل من ال bits وال bytes ولا شىء سوى ذلك ولكن على النقيض البرامج هى التى تتولى معرفة الأشياء مثلا محررات النصوص هى عبارة عن ملفات بالنسبة للينوكس ولكن محرر النصوص هو من صُمم لكى يتعامل مع الملفات الكتابية وكيفية حفظها والتعديل عليها إلخ من هذه الأمور ، كذلك وصلات الشبكة المختلفة يتعامل لينوكس معها على أساس أنها ملفات مكونة من bits و bytes ولكن تطبيقات الانترنت هى من تعلم كيفية التعامل مع الوصلات الشبكية وهكذا .

-العنصر الثانى وهو الطول الأقصى لأسماء الملفات على أنظمة اللينوكس -

لنرجع قليلا بذاكرتنا إلى الخلف حين كنا نستخدم نظام Ms-Dos ولنتذكر كم كان أقصى طول للملفات ؟ الإجابة بسيطة جدا وهو ثمانية أحرف بالإضافة إلى ثلاثة أحرف هى امتداد ونوع الملف إذا كان نصى يأخذ الشكل txt. وإذا كان فيديو مثلا يأخذ الشكل wmv. وهكذا ولكى تلاحظ ذلك مثلا أنت تستخدم ويندوز قم بتسمية ملف نصى جديد بأى اسم ولكن راعى أن يكون الإسم طويلا مثل communication theory وتأكد من عدد أحرف الكلمة ستجدها عشرون حرفا (البعض يندهش يقول هى تسعة عشر حرفا) صحيح عدد الحرف تسعة عشر حرفا ولكن لو تمعنا قليلا سوف نجد أننا قمنا بتحرير مسافة بين كلمة communication وكلمة theory ونظام التشغيل يتعامل معها على أساس أنها حرف من حروف الكلمة . والآن قم بعمل إعادة تشغيل الجهاز والدخول إلى نظام التشغيل Ms-Dos عن طريق قرص اقلاع ذاتى أو قرص مرن . لنفرض الآن أنك على محث الأوامر الخاص بال Dos ويأخذ الشكل التالى مثلا <\:A بعد ذلك قم بالدخول إلى القسم الذى قمت بإنشاء الملف النصى عليه أولا قم بكتابة اسم القسم وليكن :E ثم اضغط زر التأكيد Enter ثم قم باستعراض الملفات الموجودة على القسم باستخدام الأمر dir . الآن مثلا ظهرت الملفات الفرعية على القسم ماذا تلاحظ ؟ ...... سوف تلاحظ أن ال Dos قام باختزال اسم الملف من communication theory إلى الشكل التالى commun~1.txt وظهرت علامة جديدة هى ~ وتفيد بإن هناك تكملة للإسم ، أما أنظمة Pre-Os X Macs فالطول الأقصى لأسماء الملفات فيها كبير نسبيا بالمقارنة بنظام Ms-Dos وهو واحد وثلاثين حرفا (31) .

أما عند الحديث عن أنظمة اللينوكس واليونكس بصفة عامة فحدث ولا حرج تستطيع الطرفية أن تتعامل مع أسماء ملفات طولها الحرفى يصل إلى 255 حرف ويعتبر طول مناسب جدا لوصف ما يحلو لك لأسماء ملفاتك عند التعامل معها من خلال الطرفية !

ولكن من المفضل عند كتابة أسماء الملفات ألا يزيد طول الإسم على ثمانون حرفا كى يسهل عليك الأمر عند التعامل معه من خلال الطرفية حتى لا يحدث انكسار للاسم فى سطر آخر هذه نصيحة فضلا لا أمراً أمامك كل الخيارات مئتان فما أعلى !!!

-العنصر الثالث وهو حساسية الاسماء تجاه الحروف الكبيرة والصغيرة او (Captial and Small) -

اختلافا مع الويندوز وأنظمة الماك لينوكس عالم آخر !! نعم ففى هذا العنصر تجد دقة لينوكس المتناهية عند تحديد الأسماء والمسميات للملفات على النظام بمعنى أن لينوكس يتعامل بحساسية مفرطة مع أسماء الملفات التى تحتوى حروفا كبيرة وصغيرة ولفهم ذلك مثال :

يوجد لدينا ثلاثة ملفات بنفس الإسم (لغويا) وهم على الترتيب :

كود PHP:

```
*mysql.txt
*Mysql.txt
*MySQL.txt
```

فى المثال نجد ثلاثة ملفات نصية تحمل نفس الإسم من جهة اللغة ولكن من جهة لينوكس فالأمر مختلف جدا ، يتعامل لينوكس مع هذه الأسماء الثلاثة على أنها ثلاثة ملفات مختلفة كل منها لا يمت للآخر بصلة ، أما لو كنت على أحد أنظمة التشغيل الأخرى مثل ويندوز وقمت مثلا بإنشاء الملف الاول mysql.txt وهممت بإنشاء الملف الثانى Mysql.txt سوف يقوم النظام بإعطاء رسالة تظهر أمامك مفادها " هذا الملف موجود مسبقا هل تريد وضع الملف الجديد مع استبداله بالملف القديم ؟ "

وأما الحساسية فأيضا تعنى أن كل شىء بلينوكس مختلف إذا تضّمن حروفا كبيرة وصغيرة ، بمعنى الأمر ls يختلف تماما عن Ls يختلف عن LS وهكذا قيس الأمور كلها على نفس هذا المنوال ولذلك يجب مراعاة هذه النقطة عند كتابتك لأمر ما أو ملف ما أن تقوم بكتابته بالشكل الصحيح حتى لا يحدث مالا يحمد عقباه مثلا قد تنوى حذف ملف معين كما ذكرنا بالأعلى مثل ملف mysql.txt ولكن عن طريق الخطأ قمت بإدخال اسم ملف آخر وهو Mysql.txt حينها سيستجيب لك النظام ويفعل ما تأمره به !! لأنك حينئذ ٍالسيد الآمر مملوكه بفعل مايريد تنبه !!!

ولذلك من إحدى مميزات لينوكس أنه يعلمك الحذر فى كل ما تفعله ، وأن تكون دقيقا فى كل أمورك صغيرة كانت أو كبيرة والمثل المضروب بذلك " الوقاية خير من العلاج " ! وذاك شىءٌ جيد مقارنةً بأى نظام طبيخ آخر P:

-أما عن العنصر الرابع وهو علامات خاصة تتجنبها عند كتابتك لأسماء ملفاتك -

فنقول بحمدالله كل نظام تشغيل يحتوى على مجموعة من العلامات الخاصة تكون محجوزة للنظام فقط ولا يحق لأى مستخدم أن يقوم بوضعها فى أسماء الملفات على سبيل المثال عند تسمية ملف على نظام التشغيل ماك فلا يحق للمستخدم وضع(:) فى الأسم أو colon ،بالمثل مستخدمى ويندوز لايحق لهم وضع ال (\) أو ال backslash،وبالتالى ينطبق هذا الأمر على لينوكس كونه لا يعدو نظام تشغيل له خصوصياته ومن أهم العلامات الخاصة المحجوزة للينوكس هى ال (/) أو ال forward slash.

ولكى نستطيع فهم ذلك نضرب مثالا : نفترض أنك تريد استعراض محتويات ملف يحمل الإسم التالى books/to_buy.txt ستقوم بتنفيذ الأمر التالى :

كود PHP:

```
less /home/oem/books/to_buy.txt
```

على الفور ستظهر هذه الرسالة فى الطرفية مفادها :

كود PHP:

```
/home/oem/books/to_buy: No such file or directory
```

ولكن لماذا حدث ذلك مع العلم أننا قمنا بكتابة اسم الملف بطريقة صحيحة ؟

الإجابة حينما قمت أنت بكتابة اسم الملف المحتوى على ال/ قام الشل بالبحث عن اسم الملف بالترتيب من خلال المسار المعطى أولا فى المجلد /home ثم المجلد /oem وجاء ليقوم بفتح الملف وجد مجلد آخر اسمه /books غير موجود بالمسار لأن علامة / بينت للطرفية أن ال to_buy ملف يندرج تحت مجلد آخر اسمه /books .

من ضمن العلامات الخاصة التى لا يحق لمستخدم لينوكس استخدامها هى ال (-) أو ال dash فى بداية أسماء الملفات حيث أن استخدام هذه العلامة يحول الملف من ملف عادى ليكون فى صيغة جديدة وهى استخدام الملف كا option لأمر تقوم بتنفيذه من خلال الطرفية ولاستيعاب ذلك نفترض التالى :

تحاول إنشاء ملف جديد باسم -books_to_buy.txt وبالفعل قمت بانشاءه ولكن لسبب ما أردت حذفه لأنك لست فى حاجة إليه فمن الطبيعى أن تقوم باستخدام الأمر rm وهو الأمر الخاص بحذف الملفات وبالتالى ستكون صيغة الأمر التالى كالتالى :

كود PHP:

```
rm -books_to_buy.txt
```

سيظهر لك التالى فى الشل :

كود PHP:

```
rm: invalid option -- b
```

ومن ضمن المحظورات أيضا عند تسمية الملفات تجنب وضع مسافات بين كل كلمة فى اسم الملف مثلا يوجد الملف books to buy.txt أى يوجد مسافة بين كلمة books وكلمة to كذلك يوجد مسافة بين كلمة to وكلمة buy فلو أردت مثلا حذف هذا الملف فسيحدث خطأ لا تتوقعه حيث أن الطرفية تتعامل مع المسافات وكأنها arguments مختلفة فعند حذف الملف ستقوم بتنفيذ الأمر التالى :

كود PHP:

```
rm books to buy.txt
```

ستتعامل الطرفية بالترتيب مع الأمر لتنفيذه كالتالى :

أولا ستحاول الطرفية حذف الملف books ثم الملف to ثم الملف buy.txt وبالتالى قد يحدث مالا تتوقعه وتقوم بحذف ملفات أخرى لا تريد حذفها ووقتها لا يفيد الندم على ما فات تنبه لذلك !!! :

كود PHP:

```
rm: cannot remove `books': No such file or directory
```

```
rm: cannot remove `to': No such file or directory
rm: cannot remove `buy': No such file or directory
```

كما رأيت حاول الأمر rm حذف الملفات بالترتيب ولكن لا جدوى !!

ولكن سؤال يتبادر للأذهان وهو كيفية حذف ملف يحتوى اسمه على كلمات بينها مسافات ؟؟

يأتى هنا دور ال \ أو ال backslash حيث تقوم هذه العلامة باهمال المسافة الموجودة بين كل كلمة من كلمات اسم الملف ولتوضيح ذلك ، من المثال السابق تريد حذف الملف books to buy.tx ستقوم بعمل التالى فى الشل :

كود PHP:

```
rm books to buy.txt
```

وسيكون كل شيء على ما يرام ولكنها طريقة مرهقة فلذلك من الأفضل عدم وضع مسافات بين كلمات أسماء الملفات .

ولكن ألا توجد طريقة افضل من تلك ال \ ؟

بالفعل توجد طريقة أخرى نوعا ما أسهل لحذف ملفات تحتوى أسماؤها على مسافات بين كل كلمة وكلمة وهو وضع اسم الملف المراد حذفه بين علامتى
" " أى quotation marks مثال :

كود PHP:

```
rm "books to buy.txt"
```

ولكن مع مرور الوقت ستشعر بالإرهاق الشديد لكثرة استخدامك لل quotation makrs ولذلك من الأفضل كما أسلفنا عدم وضع مسافات سيكون خيار جيد أليس كذلك ؟!

وبعد أن استعرضنا بعض العلامات الخاصة بنظام لينوكس والتى لا يمكن استخدامها عند تسمية الملفات ، يكون كل شيء متاح لك فى التسمية بدأ من استخدام حروف فقط سواء كانت حروف صغيرة أو كبيرة ، أو حروف مع أرقام ، أو حتى أرقام فقط .

-أما عن العنصر الخامس فهو ماهى ال WildCards وما وظيفتها ؟ -

تعال لنتخيل سويا أنك تمتلك ملف نصى و مجموعة من الصور لمناظر طبيعية خلابة من صنع الملك سبحانه وتعالى أشجار وجبال وسماء وأنهار ومافى الطبيعة من جمال ربانى وهذه الصور مثلا تأخذ ترتيبا رقميا تصاعدى وتريد مثلا حذف هذه الصور نظرا لوجود نسخة أخرى منها على القرص الصلب فكيف يمكنك عمل ذلك من خلال الطرفية ؟

الإجابة يقينا ستقوم باستخدام أمر الحذف rm، ولكن هل ستقوم بحذف الصور صورة صورة أو دفعة واحدة ؟

طبعا من الممكن أن تقوم بحذف الصور عن طريق الواجهة الرسومية باستخدام الفأرة وذلك بعمل ضغطة يمنى على المجلد الموجود فيه الصور وتختار كلمة Delete، ولكن كيف لك أن تقوم بذلك من خلال الطرفية ؟

هنا يأتى دور ال Wildcards فى تنفيذ أمر ما على مجموعة ملفات دفعة واحدة وحينها لن تحتاج لتنفيذ الأمر أكثر من مرة ، وال Wildcards عبارة عن علامات خاصة أيضا مكونة من :

1- * أو ال asterisk
2- ? أو ال question mark
3- [] أو ال square brackets

والآن نستعرض كيفية الإستفادة من ال Wildcards :

على سبيل المثال الصور تاخد الترتيب التالى :

كود PHP:

```
mido1.jpg
mido2.jpg
mido3.jpg
mido4.jpg
mido5.jpg
mido6.jpg
mido7.jpg
mido8.jpg
mido9.jpg
mido10.jpg
mido11.jpg
mido12.jpg
mido1.txt
```

السؤال الأول : حذف الصور بالكامل ؟

نقوم باستخدام ال wildcard (*) لإهمال تأثير ما يأتى بعدها كالتالى :

كود PHP:

```
rm mido*.jpg
```

هنا قام الأمر بحذف الصور كاملة ولكن مع الأخذ فى الاعتبار أننا قمنا بكتابة الإمتداد حتى لا نقوم بحذف الملف mido1.txt

السؤال الثانى : حذف الصور من رقم 10-12 ؟

كود PHP:

```
rm mido1*.jpg
```

هنا قام الأمر بحذف الصور المرقمة من 10-12 مع الأخذ فى الإعتبار أننا قمنا بكتابة الإمتداد حتى لا نقوم بحذف الملف mido1.txt

السؤال الثالث : حذف الصور بما فيها الملف النصى ؟

كود PHP:

```
rm mido*
```

السؤال الرابع : حذف الملف النصى فقط ؟

كود PHP:

```
rm *.txt
```

السؤال الخامس : حذف المجلد بالكامل ؟

كود PHP:

```
rm *
```

كما رأينا كانت هذه بعض استخدامات ال * فى تنفيذ أمر عدة مرات بدون تكرار الأمر ونلاحظ السرعة فى تنفيذ الأوامر ولكن يجب الحذر حتى لا نقع فى أخطاء تسبب مشاكل !!

أما عن ال wildcard الثانية وهى ال ? فلتوضيح كيفية استخدامها نقوم بعمل التالى :

السؤال الأول : حذف الصورة الرقمة من 10-12 ؟

كود PHP:

```
rm mido1?.jpg
```

السؤال الثانى : حذف الصور المرقمة من 1-9 ؟

كود PHP:

```
rm mido?.jpg
```

لاحظ هنا الفرق مابين ال ? ، * أن ال ? تقوم بحذف خانة واحدة بعدها فقط وليس كل ما يليها ولذلك الصور من 10-12 لن تكون فى نتائج الحذف (جرب ذلك بنفسك)

السؤال الثالث : حذف الصور كلها بما فيها الملف النصى ؟

كود PHP:

```
rm mido?.*
```

هذه كانت أمثلة على استخدام ال ? فى تنفيذ أمر عدة مرات بدون تكرار أيضا مع ملاحظة الفارق بينها وبين ال* فى حالة السؤال الثانى.

والآن ننتقل إلى العلامة الثالثة والأخيرة وهى ال [] أو ال square brackets والتى تستطيع من خلالها تنفيذ أمر عدة مرات بدون تكرار ولنرى التالى :

السؤال الأول : حذف الصور 11 ، 12 فقط ؟

كود PHP:

```
rm mido1[12]
```

السؤال الثانى : حذف الصور من 10 - 12 ؟
كود PHP:

```
rm mido1[0-2]
```

وهكذا لتحديد خيارات الحذف أو الصور المراد حذفها يتم وضعها داخل ال []

وبعد أن استعرضنا بعض الأساسيات التى من الممكن احتياجها فى وقت ما سوف نستعرض أشياء أخرى تعتبر أساسية فى التعامل مع الطرفية وقد نقلتها من كتاب linux admin للكاتب أحمد نجيب الشهاوى لدعم الموضوع

أهم المفاتيح للتعامل مع الشل

space : به يتم الفصل بين الأمر والخيار الذى يتبعه .

enter : هذا المفتاح هو الذى يخبر الشل أنك انتهيت من كتابة الأمر لكى تبدأ فى التعامل مع الأمر .

backspace : عند كتابة أمر أو خيار خطأ يتم به مسح الحروف الخاطئة .

tap : يعتبر هذا المفتاح من أهم المفاتيح، فهو يظهر خاصية من أهم خصائص الطرفية ألا وهى إكمال الأمر .

مثلا إذا أردت الأمر mkdir فما عليك إلا كتابة الأحرف mkd ومن ثمّ ستكمل لك الطرفية بقية الأمر إذا ضغطت tab مرتين .

أما إن كان هناك أكثر من أمر يبدأ ب mkd (مثلا mkdev و mkdir) فسيتم عرضهم أمامك لكى تختار الأمر المناسب

ويلاحظ أنه إذا تم الضغط على ال بدون كتابة أى أحرف فسيعرض عليك النظام عرض كل الأوامر الموجودة tab به (يمكنك تجربة هذه الملاحظة) .

shift+page up : عند امتلاء الشاشة أمامك وأردت إسترجاع الجزء الذى اختفى بالأعلى فيمكننا بهذه المفاتيح رؤيتها .

shift+page down : عكس الذى قبله .

up or down : الأسهم أعلى وأسفل ، هذه أيضا تُظهر خاصية من الخصائص الهامة لل shell فال up arrow يأتى لك ثانية بالأمر السابق دون الحاجة لإعادة كتابته ثانية .

ctrl+alt+F1 : إذا كنت تعمل على الواجهة الرسومية وأردت الانتقال الى الواجهة النصية أى الشاشة السوداء، فبهذه المفاتيح الثلاثة يمكنك الانتقال اليها .

(هناك 6 شاشات consol تبدأ من F1 وحتى F6 ، أما F7 فترجع بك ثانية إلى الواجهة الرسومية)

طبعا دى كانت مقدمة بسيطة عن بعض العناصر والأساسيات اللى لازم نعرفها خلال فترة معرفتنا بسطر الأوامر وإن شاء الله فى الفصل القادم هنستعرض الأوامر الأساسية اللى تقريبا بيحتاجها مستخدم لينوكس فى شغله العادى .

طبعا الموضوع طويل إن شاء الله وهيآخد وقت

وإن شاء الله أكمل السلسلة فى مشاركات أخرى فا صبرا عليا شوية علشان الموضوع يخرج فى أحسن صورة

السلام عليكم ورحمة الله وبركاته

السلام عليكم ورحمة الله وبركاته

عدنا لِنُكمل المسيرة الشيقة مع سطر الأوامر ونبدأ بسم الله :

الجزء الثانى : الأوامر الأساسية فى لينوكس

استعرضنا فى الفصل السابق بعض الأساسيات التى سيتم تناولها على مدار الموضوع وفى هذا الفصل سنتعرض لبعض الأوامر الأساسية والمهمة لمستخدم لينوكس بشكل يومى وسيتم شرحها بالتفصيل إن شاء الله .

الأمر : ls

يعتبر الامر ls من أكثر الأوامر شيوعا وانتشارا واستخداما لما له من أهمية شديدة لمستخدم لينوكس بشكل شبه يومى . ولمعرفة أهمية الأمر نفترض أنك تمتلك ملفات تخصك على احدى المجلدات وفى يوم ما أردت استعراض هذه الملفات على هذا المجلد فما الذى يمكّنك من ذلك ؟

الإجابة هو الأمر ls حيث يقوم الأمر باستعراض الملفات والمجلدات الفرعية تحت المجلد الرئيسى التى توجد فيه هذه الأشياء ، والآن لنبدأ سويا رحلة البحث عن المتعة باستخدام سطر الأوامر ، لمستخدمى واجهة جنوم عليهم الذهاب للمسار التالى :

كود PHP:

```
Applications => Accessories => Terminal
```

والآن على الفور أول شىء نقوم بكتابة الأمر ls فى صورته البسيطة دون دمجه مع أى خيارات اضافية ثم بعد ذلك نضغط على الزر enter كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls
1.rmvb
38AR_Crucifiction_Fact_or_Fiction_vs_Douglas_512KB.rmvb
Desktop
documentation and presentation of parallel processing for SDR us
ing OMAP.rar
Examples
w32codecs_20061022-0.0_i386.deb
```

كما يمكنك استعراض الملفات الموجودة على مجلد فى المسار الحالى دون الحاجة إلى كتابة مسار المجلد ولنفترض أننا نريد استعراض محتويات المجلد Desktop نقوم بعمل التالى :

كود PHP:

```
oem@ubuntu:~$ ls Desktop
33391_hi.ram
4.rm                       index_files
index.php                  spraah2.pdf
```

كما يمكنك أيضا استعراض نفس المجلد عن طريق المسار المرتبط الخاص به أو ما يسمى ال relative path وذلك كالتالى :

كود PHP:

```
oem@ubuntu:~$ ls /home/oem/Desktop/
33391_hi.ram                                spraah2.pdf
4.rm                                        index_files
index.php
```

ولكن ألا توجد طريقة أسهل فى التعامل لعرض ملفات ال home directory تكون أفضل من ذلك ؟

نعم ، توجد العلامة ~ والتى تقوم نفس مقام المسار /home/oem دون أدنى اختلاف ولنتأكد من ذلك من خلال الطرفية بعمل التالى :

كود PHP:

```
oem@ubuntu:~$ ls ~/Desktop
33391_hi.ram                                spraah2.pdf
4.rm                                        index_files
index.php
```

ونلاحظ التالى أن الأمر ls قام باستعراض الملفات الموجود على المجلد الحالى أى المسار الفعلى الذى نكون عليه لحظة كتابة الأمر أو أين من المجلدلت الفرعية على المسار ، ولننتقل إلى نقطة أخرى ألا وهى : هل يتطلب أن نكون على المسار كى نعرض محتوياته؟

الإجابة بالطبع لا فقد تكون على مسار وتقوم باستعراض ملفات مسار آخر ولاستعياب ذلك نفترض أننا على مسار /home/oem ونريد استعراض ملفات مسار آخر فلنرى ذلك فى الطرفية :

كود PHP:

```
oem@ubuntu:~$ pwd
/home/oem
```

لاحظ أننا أولا قمنا بمعرفة المسار الحالى عن طريق استخدام الأمر pwd لمجرد التأكد ليس إلا .

كود PHP:

```
oem@ubuntu:~$ ls /var/cache/
apt   debconf             gnome-system-tools  man         setu
p-tool-backends
cups  dictionaries-common  locate             pppconfig
```

كما رأيت فعلنا ذلك بكل سهولة ويسر ، ولننتقل إلى نقطة أخرى وهى كيفية ايجاد ملف معين على مجلد معين على مسار معين كيف لك أن تقوم بذلك ؟

أوضح قليلا الآن حان دور استخدام ال wildcards. نفترض أننا نود البحث عن ملف اسمه mysql_unde_rground.pdf وكل ما نتذكره عن الملف اسمه والامتداد الخاص به والمجلد الذى يوجد به ولكن المجلد يحتوى على ملفات ذات الامتداد pdf. وأخرى ذات امتداد txt.وما نريده هو عرض الملفات التى تنتهى بامتداد pdf.نقوم بعمل التالى :

كود:

```
oem@ubuntu:~$ ls ~/Desktop/*.pdf
/home/oem/Desktop/mysql_under_ground.pdf  /home/oem/Desktop/spraah2.pdf
/home/oem/Desktop/new.pdf
```

الآن ظهرت أهمية استخدام ال wildcards كما رأينا ، ولكن نحدد عنصر البحث فى استعراض الملف بواسطة جزء من اسمه نقوم بعمل التالى :

كود:

```
oem@ubuntu:~$ ls ~/Desktop/*under*.pdf
/home/oem/Desktop/mysql_under_ground.pdf
```

بعد أن ظهرت أهمية ال wildcards وكيفية استخدامها ودمجها مع الأمر ls نبدأ الآن فى استعراض الخيارات التى يمكن اضافتها مع الامر .

1- استعراض الملفات الموجودة على المجلدات الفرعية :

كود PHP:

```
View a List of Files in Subfolders (ls -R or ls --recursive)
```

يعتبر الخيار R- من ضمن الخيارات المهمة والتى يتم إضافتها إلى الأمر ls وذلك لاستعراض الملفات الموجودة على المجلدات الفرعية داخل مجلد رئيسى بسهولة ويسر دون الاحتياج إلى فتح كل مجلد على حده كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -R ~/Desktop
/home/oem/Desktop:
33391_hi.ram                                 index.php
4.rm                                         mysql_under_ground.p
df
new.pdf                                      spraah2.pdf
index_files

/home/oem/Desktop/index_files:
362922962_1887f6723a_o.png  ips_menu_html.js
365159135_b0fcd2be76_o.jpg  ips_menu.js
7951.png                    ips_xmlhttprequest.js
Alfred.gif                  js.js
```

```
arab.                           jss.php
arab_data                       Market.jpg

/home/oem/Desktop/index_files/arab_data:
arab_01.gif  arab_04.gif  arab_10.gif  arab_13.gif
arab_02.gif  arab_08.gif  arab_11.gif  arab_14.gif
arab_03.gif  arab_09.gif  arab_12.gif  arab_15.gif
```

كما رأينا أعلى قام الأمر أولا باستعراض الملفات بالمجلد الرئيسى Desktop ثم بعد ذلك تم استعراض الملفات الموجودة داخل المجلد الفرعى index_file ثم بعد ذلك تم استعراض الملفات الموجودة داحل المجلد الفرعى arab_data ، ولاحظ أيضا أن كل مجلد فرعى يظهر بالمسار الإرتباطى الخاص به أو ال relative path.

ولكن هل الخيار R- دائما هو الأفضل ؟

الإجابة لا ، ولتعرف لماذا جرب التالى فى الطرفية وستعرف ماذا أقصد :

كود PHP:

```
ls -R /var
```

2- استعراض الملفات الموجودة على أى مجلد فى عمود رأسى واحد فقط :

كود PHP:

```
View a List of Contents in a Single Column (ls -lor ls --
format=single-column)
```

تعودنا قبل ذلك عند كتابة الأمر ls وحده دون اضافة أى خيارات أن تظهر الملفات على شكل أعمدة وصوف وبين كل عمود وعمود مسافتين لسهولة قراءة خرج الأمر ، ولكن الجديد مع الخيار 1- أنه يمكنك استعراض ملفات أى مجلد فى عمود رأسى واحد فقط كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -1 ~/Desktop
33391_hi.ram
4.rm
index_files
index.php
mmmm
spraah2.pdf
```

ملحوظة : يمكن استخدام الخيار R- مع الخيار 1- لعمل الوظيفتين اللتان يقوم بهما الخياران فى آن واحد أى استعراض الملفات على المجلدات الفرعية داخل المجلد الرئيسى مع عرض نتيجة كل مجلد فرعى فى عمود رأسى واحد فقط ليكون الأمر بالشكل التالى :

كود PHP:

```
ls -1R ~/Desktop
```

أما إذا استخدمت الخياران 1R- مع الأمر ls فى مجلد رئيسى يحتوى على مئات المجلدات الفرعية مثل /var ، فقد تحتاج إلى ctrl+c ... حينها ستعرف فائدة الضغط على هذه المفاتيح........

3- استعراض الملفات الموجودة على أى مجلد مع وضع Comma بين كل ملف وآخر :

كود PHP:

```
view Contents As a Comma-Separated List ( ls -m )
```

خيار آخر لتنظيم شكل الملفات عند استعراضها باستخدام الأمر ls وهو الخيار m- حيث يقوم بوضع comma بين كل ملف وآخر كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -m /var
backups, cache, games, lib, local, lock, log, mail, opt, run, s
pool, tmp
```

4- استعراض الملفات والمجلدات المخفية على مسار معين :

كود PHP:

```
View Hidden Files and Folders (ls -a or ls --all)
```

حتى هذه اللحظة كانت نتائج استعراض الملفات والمجلدات سواء كانت رئيسية أو فرعية باستخدام الأمر ls الملفات والمجلدات المرئية فقط ، ولكن ماذا لو كانت هناك ملفات أو مجلدات مخفية ؟ هنا الخيار a- ليقوم بهذه المهمة ، فعلى سبيل المثال مجلد ال home directory الخاص بك يحتوى ملفات مخفية والبعض بل يمكن الكثير لا يعلم ذلك ولكى تعرف ذلك تابع التالى :

كود PHP:

```
oem@ubuntu:~$ ls -a ~/
.
..
.bash_history
.bash_logout
.bash_profile
.bashrc
books~
.DCOPserver_ubuntu__0
.DCOPserver_ubuntu_:0
Desktop
.dmrc
using OMAP.rar
.esd_auth
.evolution
Examples
.fonts.cache-1
.mozilla
.mplayer
```

لاحظ : المجلدات المخفية يكون فى بدايتها نقطة أو ال dot ، وقد رأيتَ فى المجلد الرئيسى والخاص بالمستخدم الذى أنشأته أثناء مرحلة التنصيب يحتوى على الكثير من المجلدات الفرعية مع مجلدات أخرى فرعية ولكنها مخفية ، بعض من هذه المجلدات المخفية تكون لبرامج أنت تقوم بتنصيبها وتحتوى على ملفات ال configuration الخاصة بها ، والبعض الآخر لبرامج أساسية يتم تنصيبها أثناء تثبيت التوزيعة بدون تدخل من المستخدم.

أيضا من ضمن الأشياء التى أودُ لفت انتباهك إليها هو أول سطرين فى نتيجة الأمر أى ال. ، .. (dot and double dots) حيث تشير ال . إلى المجلد الحالى الذى توجد فيه الآن أما ال .. فتشيران إلى المجلد الأب الذى يحوى المجلد الرئيسى وطبيعى فى هذه الحالة هو المجلد home/ بحيث يكون المجلد home/ هو المجلد الأب أو فيما يسمى بال parent directory للمجلد oem/. هذان المجلدان متواجدان بشكل عام فى كل المسارات التى توجد بالتوزيعة ولا تستطيع التخلص منهم . وبالتالى توقع أن تَجِدْهما فى كل مرة تستخدم فيها الخيار a- . ولك أن تشكر الخيار a- أن نبهك إلى وجود ملفات ومجلدات لم تكن تعلم عنها شيئا !!

5- استعراض الملفات مع إظهار أنواعها :

كود PHP:

```
Visually Display a Files Type (ls -F or ls --classify)
```

بأعلى استعرضنا بعض الخيارات التى يتم إضافتها للأمر ls لتُكسبه رونقاً رائعا وتُضفى عليه لمسات جذابه لتعرضْ لك ما تريده ولكن هل تظن الأمر ينتهى عند هذا الحد ؟ البعض قد يظن أن الأمر ls قد أتى بكل مافى جعبته ، ولكن ليس الآن هناك المزيد والمزيد.....

من ضمن الخيارات الرائعة التى تأتى مع الأمر ls هو الخيار F- ، حيث يقوم هذا الخيار بعرض نوع الملف الموجود على المجلد سواء كان الملف تنفيذى أو Executable ، مجلد أو Direcotry ، رابط رمزى أو Symbolic link ، أو FIFO وهى اختصار لعبارة First in First out وليس هناك مجال لشرح ال FIFO لأنه يحتاج درس آخر خاص به ، كل ما يهمنى الآن أن تعرفه أن الخيار F- يقوم بعرض رموز أو علامات كل منها تدل على نوع من الأنواع التى قمت بذكرها ..... والآن لنستعرض كيفية كتابة الأمر فى الطرفية كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -F /bin
bzgrep*                     gunzip*          nano*        umount
*
bzip2*                      gzexe*           nc*          uname*
bzip2recover*               gzip*            netcat@      uncomp
ress*
bzless*                     hostname*        netstat*     vdir*
bzmore*                     ip*              pidof@       which*
mido/                       mysql/
```

كود PHP:

```
oem@ubuntu:~$ ls -F /dev
log=        ptyc9  ptyrd  ptyx1  tty18       ttybe  ttyr2   ttyt
6   ttyya
loop/       ptyca  ptyre  ptyx2  tty19       ttybf  ttyr3   ttyt
```

```
7   ttyyb
```

لاحظ : قمت فى المثال السابق باستعراض أنواع بعض الملفات للمجلد /bin ، والبعض الآخر للمجلد /dev باستخدام الخيار F- وظهرت لدىّ نتيجة الأمر كما رأيت ، تدل العلامة * على أن الملف الذى توجد أمامه أنه ملف تنفيذى أو كما قلنا Executable ، أما العلامة@ فتدل على أن الملف رابط رمزى أو Symbolic link ، أما العلامة / فتدل على أن الملف مجلد ، والعلامة= تدل على ملف من نوع Socket ولذلك فهى علامات خاصة لمعرفة نوع الملفات من بين الأنواع السابق ذكرها .

ملحوظة : المجلد /bin الشكل الإفتراضى له لا يحتوى على مجلدات نهائيا ولذلك قمت بوضع مجلدين فرعيين بداخل المجلد للتوضيح فقط ...

هذا الجدول تلخيص للكلام السابق :

كود PHP:

```
Character       Meaning

*               Executable

/               Directory

@               Symbolic Link

|               FIFO

=               Socket
```

6- استعراض الملفات ، التصاريح عليها ، المالك ، وأشياء أخرى...

كود PHP:

```
List Permissions, Ownership, and More (ls -l or ls --
format=long)
```

من هنا نبدأ العمل الحقيقى ...... أجل فمن هذه اللحظة يبدأ مستخدم الطرفية ليشعر وكأنه محترف ، تعلمنا سابقا أن الأمر ls يقوم بإظهار المجلدات الفرعية والملفات التى توجد داخل المجلدات الرئيسية بأشكال مختلفة نظرا لاستخدام خيارات مختلفة أيضا ، منها ما يُظهر نوع الملف ،ومنها ما يُظهر الملفات المخفية ، ومنها ما يقوم باستعراض الملفات فى هيئات مختلفة سواء كانت مفصولة ب Comma أو تظهر فى شكل عمود رأسى واحد وهكذا....

ولكن لم نتعرض إلى الملفات بذاتها ، هنا الخيار l- ليقوم بدور مهم جدا على الملفات ألا وهو استعراض التصاريح عليها ، استعراض المالك لها ، تاريخ الإنشاء أو التعديل ، حجم الملف وغيره من المعلومات المفيدة جدا والتى قد يستخدمها بعض المخربين أثناء العبث بأغراض الآخرين !! ومثال على ذلك لتوضيح صيغة الأمر ls وإضافة الخيار l- معه كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -l
total 972012
```

```
drwxr-xr-x 3 oem oem 364911700 2006-10-22 20:01 mysql
-rwxr-xr-x 3 oem oem 364911700 2006-10-22 20:01 addext
drwxr-xr-x 4 oem oem      4096 2007-01-27 06:26 Desktop
drwxr-xr-x 2 oem oem     4096 2007-01-25 18:24 mm
```

والخيار l- يرمز إلى الكلمة long ، وبالتالى نستنبط من معنى الكلمة long ألا وهو الطول ،أى أن الخيار يمدنا بالكثير من المعلومات المهمة جدا عن الملفات والمجلدات والآن لنستعرض كيفية فهم السطر من اليمين إلى اليسار ونأخد مثال على ذلك السطر الأول :

أولا : الإسم mysql يدل على اسم الملف الذى تم استعراضه من قِبل الأمر ls نتيجة كتابة الخيار l- معه .

ثانيا : الأرقام 20:01 ، 22-10-2006 تدل على الساعة والتاريخ اللذين تم فيهما التعديل الأخير على الملف .

ثالثا : الرقم 364911700 يوضح حجم الملف mysql ولكن بال bytes

رابعا : الإسم oem من جهة اليمين يوضح المجموعة المالكة للملف .

خامسا : الإسم oem من جهة اليسار يوضح المالك الأصلى للملف (المستخدم الذى قام بإنشاء الملف) .

سادسا : الرقم بجوار مالك الملف ، مثلا فى حالة الملف Desktop (كما ذكرنا قبل ذلك المجلد يعتبر نوع من الملفات الخاصة) فى السطر الخاص به رقم 4 وهذا الرقم له حالتان :

1- إذا كان الرقم فى سطر خاص بملف عادى فهو يوضح عدد ال hard links الموجودة لذلك الملف .

2- إذا كان الرقم فى سطر خاص بمجلد فهو يوضح عدد المجلدات الفرعية الموجودة بداخل المجلد الرئيسى وفى مثالنا المجلد Desktop يحتوى على أربع مجلدات فرعية .

ملحوظة : قمت بإنشاء مجلد فارغ وأسميته mm كما تلاحظون فى السطر الأخير ولكن يوجد فى السطر الخاص بهذا المجلد رقم 2 فكيف ذلك وقد ذكرت أن المجلد فارغ ؟

لنرجع قليلا إلى الوراء وبالتحديد فى الخيار a- فقد ذكرت أنه يوجد مجلدان فرعيان بشكل دائم مع أى مجلد رئيسى وهما ال . و ال .. اى المجلد الحالى والمجلد الأب (parent directory) ولذلك وجدنا عدد المجلدات الفرعية بالمجلد mm وقد وُضِعَ فى السطر الخاص به ولنرى ذلك :

كود PHP:

```
oem@ubuntu:~$ ls -a mm/
.   ..
```

سابعا : الرموز drwxr-xr-x وتشير إلى التصريح الخاص بالملف mysql ، ولكن من الوهلة الأولى يبدو وكأنه شفرة سرية !! ولكنه فى الحقية من الأمور السهل جداً فهمُها واستيعابها بقليل من المعرفة ، لاحظ معى هناك عشرة أحرف مقسمين إلى أربع مجموعات (قد لا يبدو الأمر كذلك) .

المجموعة الأولى : تتكون من الحرف الأول (1).

المجموعة الثانية : تتكون من الحرف الثانى إلى الحرف الرابع (2-4)

المجموعة الثالثة : تتكون من الحرف الخامس إلى الحرف السابع (5-7)

المجموعة الرابعة : تتكون من الحرف الثامن إلى الحرف العاشر (8-10)

وعلى سبيل المثال نأخذ التصريح الخاص بالملف mysql ويبدو مقسما إلى المجموعات السابق ذكرها بالشكل التالى :

كود PHP:

```
d|rwx|r-x|r-x
```

والآن تعالوا معى لنعرف بالضبط كيفية فهم هذا التصريح من اليسار إلى اليمين:

المجموعة الأولى والتي تشمل الحرف الأول (وفى حالتنا هذه هو حرف ال d) تمثل نوع الملف سواء كان مجلد (directory) أو ملف عادى (regular file) وطبعا من البديهى أن يرمز الحرف d إلى الكلمة (directory) أو مجلد ، وبالتالى يعتبر الملف mysql من نوع مجلد أو ملف خاص كما ذكرنا سابقا ، بينما العلامة- أو ال dash توضح أنه ملف .وطبعا توجد بعض الخيارات التى تؤدى هذه المهمة ولكن عن طريق الرموز الخاصة بها مثل الخيار F- والذى من ضمن علاماته ال / والتى ترمز إلى مجلد كما يمكن معرفة ذلك أيضا من خلال الخيار color-- حيث عند دمج هذا الخيار مع الأمر ls تظهر ألوان للملفات بشكل مختلف ويرمز كل لون إلى نوع معين من الملفات وأكْتفى فقط بسرد الجدول الخاص بهذه الألوان كالتالى :

كود PHP:

```
Color                              Meaning

Default shell text color           Regular File

Green                              Executable

Blue                               Directory

Magenta                            Symbolic Link

Yellow                             FIFO

Magenta                            Socket

Red                                Archive (.tar ,.zip ,.deb ,.r
pm )

Magenta                            Imgaes  (.jpg ,.gif ,.png ,.t
iff)

Magenta                            Audio   (.mp3 ,.wav ,.ogg)
```

وبالطبع يوجد الكثير من هذه العلامات التى توضح الأنواع المختلفة من الملفات والجدول التالى يوضح كلامى :

كود PHP:

Character	Meaning
-	Regular File
-	Executable
d	directory
l	Symbolic Link
s	Socket
b	Block Device
c	Character Device
p	Named Pipe

أما الأحرف التسعة الباقية فطبيعى تمثل المجموعة الثانية ، والثالثة ، والرابعة على الترتيب وبالتالى تكون التصاريح الخاصة بالملف على الترتيب تخص كلاً من المالك الأصلى أو file's owner ، المجموعة التابع لها الملف أو file's group ، والتصريح الأخير لمجموعة المستخدمين الباقين على النظام أو other users ، ففى المثال السابق نجد التصريح الخاص لمالك الملف oem هو rwx والتصريح الخاص بالمجموعة oem هو r-x والتصريح الخاص بباقى المستخدمين على النظام هو r-x ولكن ماذا تعنى هذه الحروف ؟

فى كل حالة يعنى الحرف r العبارة التالية " yes , read is allowed" أو يُسمح بقراءة الملف ; أما الحرف w فيعنى "yes , write is allowed" أو يسمح بالكتابة على الملف (وكلمة "write" تعنى كلاً من التعديل والحذف أو ال changing and deleting) أما الحرف x فيعنى " yes , execute is allowed" أو يسمح بتنفيذ الملف إذا كان Executable ، أما عن العلامة- فتعنى " no , dont allow this action" فمثلا لو أتت ال- مكان الحرف r فى تصريح أى مجموعة فوقتها تعنى "no , read is not allowed" أو تصريح القراءة غير مسموح به كذلك لو أتت هذه مكان الحرف w أو الحرف x.

وبالنظر إلى الملف addext مثلا نجد أن المالك oem له تصريح rwx أى يستطيع القراءة والكتابة والتنفيذ على الملف ، أما المجموعة oem فلها التصريح r-x أى تستطيع القراءة والتنفيذ فقط ، وباقى المستخدمين لهم التصريح r-x أى يستطيعوا أيضا القراءة والتنفيذ فقط . هذا الكلام ينطبق على الملفات حيث معنى القراءة والكتابة مفهوم ولكن ألا يوجد شىء غريب بالموضوع كيف لنا مثلا أننا ننفذ مجلد كما فى حالة المجلد mysql ؟

قلنا أن كلاً من الحرفين w ، r سواء بالنسبة للمجلد أو الملف العادى يحتملان نفس المعنى ولكن الوقفة هنا مع الحرف x حيث يختلف تنفيذ الملف عن تنفيذ المجلد كيف ذلك ؟

الحرف r إذا جاء فى تصريح خاص بمجلد فهو يعنى أنه يمكنك تصفح محتويات المجلد دون عوائق ، أما إذا جاء الحرف فى تصريح ملف فكما قلنا آنفا أنه يمكنك قراءة الملف ، أما بالنسبة للحرف w إذا جاء فى تصريح مجلد فهو يعنى أنه يمكنك اضافة محتويات أخرى للمجلد ، تسمية محتويات موجودة بأسماء

أخرى ، حذف اى ملف من ملفات المجلد ،

أما إذا جاء الحرف w فى تصريح ملف فهو يعنى أنه يمكنك الكتابة على هذا الملف والتعديل عليه باحدى محررات النصوص المعروفة مثل nano او vi وغيرها .

والآن حان دور الحرف x فإذا جاء الحرف x فى تصريح مجلد فهو يعنى أنه يمكنك الولوج إلى ذلك المجلد وتنفيذ البرامج الموجودة بداخله ، أو الولوج إلى المجلدات الفرعية التى توجد بداخله ، أما إذا جاء الحرف فى تصريح ملف فهو يعنى انه يمكنك تنفيذ الملف ليقوم بعمل مهمة او أداء وظيفة معينة أو بمعنى " Execute the file to become a program".

والآن وكما تعلمنا مدى قوة الخيار l- والتى ظهرت بشكل ٍجَلِىّ فى الأمثلة السابقة ، ولكن تزداد قوة الخيار l- أكثر وأكثر عند دمجه مع خيارات أخرى مثل الخيار a- والذى تعلمنا أيضا قوته فى إظهار الملفات المخفية والتى قد لا نلقى لها بالاً ليكون ناتج الامر ls فى النهاية بشكل التالى بعد دمج الخيارين :

كود PHP:

```
oem@ubuntu:~$ ls -al ~
drwxr-xr-x  3 root root      4096 2007-01-22 02:12 .
drwxr-xr-x 31 oem  oem       4096 2007-01-28 21:57 ..
-rw-r--r--  1 oem  oem   18164916 2007-01-28 06:05 A3.41.zip
-rw-------  1 oem  oem       6260 2007-01-29 08:33
.bash_history
rw-r--r--  1 oem  oem       2227 2007-01-22 02:12 .bashrc
-rw-------  1 oem  oem          0 2007-01-23 11:50 books~
drwxr-xr-x  9 oem  oem       4096 2007-01-29 10:23 Desktop
```

كما ترون قمنا باستعراض الملفات الموجودة على ال home directory أو ~ باستخدام خياران يقومان بوظيفتين مختلفتين وهما على الترتيب :

* استعراض الملفات المخفية والتى تخص الخيار a-

* استعراض الملفات ، التصاريح عليها ، المالك ، وأشياء أخرى... والتى تخص الخيار l-

7- استعراض الملفات مع عكس الترتيب الهجائى لها

كود PHP:

```
Reverse The Order Contents Are Listed (ls -r or ls --reverse)
```

لو تمعنا قليلا فى الخيار السابق ألا وهو l- ودققنا النظر لوجدنا أنه يقوم بسرد الملفات ويقوم بترتيبها ترتيباً هجائياً بمعنى حسب ترتيب الحروف باللغة الإنجليزية يبدأ أولا بالملفات التى تحتوى فى بداية أسمائها على الحرف A ، ثم الملفات التى تحتوى على الحرف B ، ثم الملفات التى تحتوى على الحرف C ، وهكذا ولكن ماذا لو أردنا أن نقوم بعمل العملية العكسية ، بمعنى أوضح نستعرض الملفات التى تحتوى فى بداية أسمائها على الحرف z ، ثم الحرف y ، ثم الحرف x ...... انتهاءً بالحرف A ؟

هنا الخيار r- ليقوم بتلك المهمة حيث يقوم باستعراض الملفات ترتيباً هجائياً معكوسا وبالتالى يتم دمج الخيار r- مع الأمر ls كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -alr ~

drwxr-xr-x  9 oem  oem       4096 2007-01-29 10:23 Desktop
-rw-------  1 oem  oem          0 2007-01-23 11:50 books~
rw-r--r--  1 oem  oem       2227 2007-01-22 02:12 .bashrc
-rw-------  1 oem  oem       6260 2007-01-29 08:33
.bash_history
-rw-r--r--  1 oem  oem   18164916 2007-01-28 06:05 A3.41.zip
drwxr-xr-x 31 oem  oem       4096 2007-01-28 21:57 ..
drwxr-xr-x  3 root root      4096 2007-01-22 02:12 .
```

لاحظ : كما أسلفنا سابقا وذكرنا أن لينوكس حساس تجاه الحروف ال capital والحروف ال small ولذلك الخيار r- مختلف تمام عن الخيار R-

8- استعراض الملفات وتصنيفها عن طريق الإمتداد الخاص بها

كود PHP:

```
Sort Contents By File Extension (ls -X  or ls --sort=extension)
```

يُمكنك الخيار X- من استعراض الملفات وترتيبها ترتيبا هجائيا ولكن عن طريق الامتداد الخاص بها ، بمعنى نفترض أنه يوجد لديك أربعة ملفات بامتدادات مختلفة على سبيل المثال :

كود PHP:

```
keymouse.zip
mido.rar
mysql.txt
image.jpg
```

فحينما تقوم بتنفيذ الأمر ls -X على المجلد الذى توجد فيه هذه الملفات ، ولنفترض أنه المجلد ~ أو ال home directory فسيقوم الأمر بترتيب الملفات كالتالى :

كود PHP:

```
oem@ubuntu:~$ ls -X
image.jpg
mido.rar
mysql.txt
keymouse.zip
```

لاحظ : ترتيب الحروف الهجائية للغة الإنجليزية للملفات الأربعة أو الحرف j ، ثم الحرف r ، ثم الحرف t ثم أخيرا الحرف z .

وبالطبع تستطيع دمج الخيار X- مع الخيار l- ليعرض لك الكثير من المعلومات عن الملفات وفى نفس

الوقت يقوم بترتيبهم حسب الإمتدادات الخاصة بهم كما يلى :

كود PHP:

```
drwxr-xr-x 9 oem oem      4096 2007-01-29 10:23 Desktop
drwxr-xr-x 2 oem oem      4096 2007-01-27 07:50 mm
-rw-r--r-- 1 oem oem  18164916 2007-01-28 06:05 A3.41.tar.gz
-rw-r--r-- 1 oem oem    183457 2007-01-23 15:37 gg.mht
-rw-r--r-- 1 oem oem    647142 2007-01-25 11:05 Firefox_wallpap
er.png
-rw-r--r-- 1 oem oem 364911700 2006-10-22 20:01 1.rmvb
-rw-r--r-- 1 oem oem   5654152 2004-01-01 05:46 IAA_2.2.0.215
0.zip
```

ملحوظة : عندما توجد مجلدات فرعية داخل المجلد الرئيسى فتكون فى مقدمة الترتيب لأنها طبيعى لا تحتوى على امتدادات ، ولاحظ أيضا إنه عندما توجد ملفات تحتوى أكثر من امتداد كما فى حالة الملف A3.41.tar.gz فيكون الترتيب خاص بالإمتداد النهائى ألا وهو gz ويكون الترتيب على أساسه .

9-استعراض الملفات وتصنيفها عن طريق الأحجام الخاصة بها

كود PHP:

```
Sort Contents By Size (ls -S or ls --sort=size)
```

تستطيع أيضا استعراض الملفات عن طريق الأحجام الخاصة بها على القرص الصلب لديك باستخدام الخيار S- وإضافته للأمر ls كما يلى :

كود PHP:

```
oem@ubuntu:~$ ls -laS ~

-rw-r--r--  1 oem  oem  364911700 2006-10-22 20:01 1.rmvb
-rw-r--r--  1 oem  oem   18164916 2007-01-28 06:05 A3.41.zip
-rw-r--r--  1 oem  oem    5654152 2004-01-01 05:46 IAA_2.2.0.21
50.zip
-rw-------  1 oem  oem    1962871 2007-01-24 12:23 ghhh
-rw-r--r--  1 oem  oem     647142 2007-01-25 11:05 Firefox_wall
paper.png
```

لاحظ : قمت بإضافة الخيار S- إلى الخيارين l- و a- لبيان وظفيته بشكل واضح ، كما أود لفت انتباهك إن أحجام الملفات التى يقوم الخيار S- بإظهارها تكون بال bytes .

10-استعراض الملفات وتصنيفها عن طريق الأحجام الخاصة بها فى صورة KB ، GB ، MB

كود PHP:

```
Express File Sizes in Terms of KB , MB , GB (ls -h or ls --
human-readable)
```

ملحوظة :

كود PHP:

```
KB = kilo byte

MB = mega byte

GB = giga byte
```

استعرضنا فى الخيار السابق كيفية عرض الملفات باستخدام الأمر ls وتصنفيها مستخدمين الخيار S- ولكن كما ذكرت كانت الأحجام المعروضة بال byte فقط ، ولذلك أتى الخيار h- ليكون مُكملا للخيار S- حيث يقوم بعرض أحجام الملفات بصورة أفضل من سابقتها وهى عرضها فى صورة mega ، giga byte ، giga byte ، byte ولنرى ذلك بالمثال التالى :

كود PHP:

```
oem@ubuntu:~$ ls -laSh ~

-rw-r--r--  1 oem  oem  349M 2006-10-22 20:01 1.rmvb
-rw-r--r--  1 oem  oem   18M 2007-01-28 06:05 A3.41.zip
-rw-r--r--  1 oem  oem  5.4M 2004-01-01 05:46 IAA_2.2.0.215
0.zip
-rw-------  1 oem  oem  1.9M 2007-01-24 12:23 ghhh
-rw-r--r--  1 oem  oem  632K 2007-01-25 11:05 Firefox_wallpape
r.png
-rw-r--r--  1 oem  oem  437K 2007-01-24 09:10 parallel processi
ng for SDR using OMAP.rar
-rw-r--r--  1 oem  oem  196K 2007-01-29 08:01 .xsession-errors
```

لاحظ أننى قمت بإضافة الخيار h- إلى خيارات أخرى لكى نحصل على نتائج أكثر فعالية .

وبذلك نكون قد انتهينا من شرح معظم الخيارات المهمة والتى يمكن إضافتها للأمر المهم ls حتى نحصل على نتائج أكثر فعالية ومفيدة فى نفس الوقت فى تحقيق مطلوبة

السلام عليكم ورحمة الله وبركاته

الأمر : pwd

كود PHP:

```
(Display The Path of Your Current Directory)
```

وبعد أن انتهينا من الأمر ls ، سنبدأ بمشيئة الله فى الانتقال إلى مرحلة جديدة من الأوامر ، ونستهل بدايتها مع الأمر pwd .

قد تطرقت لذكر الأمر pwd سابقا خلال شرحى للأمر ls وذكرت الوظيفة التى يقوم بها ، ألا وهى عرض المسار الحالى الذى توجد فيه ، ولنقوم بتوضيح الأمور أكثر نفترض أنك قمت مثلا بتنفيذ عدة أوامر من خلال الطرفية وقمت بالولوج عدة مرات إلى مسارات مختلفة وأثناء أداء هذه المهام لم تتذكر المسار الحالى الذى توجد فيه فما العمل ؟ الحل فى الأمر pwd والذى بمجرد كتابته يعرض لك المسار على الفور مثال :

كود PHP:

```
oem@ubuntu:~$ pwd
/home/oem
```

والأمر pwd هو اختصار للعبارة التالية print working directory أو بمعنى قم بطبع الدليل الذى نعمل عليه .

لاحظ : أن كلمة print تعنى أظهِرْ الخرج على الشاشة أو بمعنى print to screen وليس بمعنى اعطاء أمر ارسال الطباعة للطابعة .

الأمر : cd

كود PHP:

```
(Change to a Different Directory)
```

تكلمنا سابقا عن كيفية استعراض محتويات أى مجلد بسهولة ويسر مُستخدمين الأمر ls وذلك بمجرد تحديد المسار الذى يوجد فيه المجلد أو كتابة ال relative path ، ولكن ماذا لو أردنا تغيير المجلد الحالى الذى نوجد فيه إلى مجلد آخر ؟

هنا الأمر cd لينجز هذه المهمة حيث يقوم الأمر بتغيير الدليل الذى نوجد عليه إلى أى دليل آخر (مسار أقصد) على التوزيعة ومثال على ذلك نفترض أننا نوجد داخل المسار التالى /home/oem/Desktop ونريد تغيير هذا المسار إلى آخر وهو /var/cache كل ما علينا فعله التالى :

نقوم أولا بالتأكد أننا على المسار الحالى :

كود PHP:

```
oem@ubuntu:~/Desktop$ pwd
/home/oem/Desktop
```

بعد ذلك نقوم بكتابة المسار الذى نريده ومن ثمّ نضغط على زر enter :

كود PHP:

```
oem@ubuntu:~/Desktop$ cd /var/cache
```

بالفعل تحولنا للمسار المطلوب كما بالشكل :

كود PHP:

```
oem@ubuntu:/var/cache$
```

وللتأكد مرة أخرى أننا على المجلد المطلوب :

كود PHP:

```
oem@ubuntu:/var/cache$ pwd
/var/cache
```

ولكن ماذا لو أردنا مرة أخرى الرجوع إلى ال home directory ؟

ذكرنا سابقا أن العلامة~ تساوى بالضبط ال home directory ولذا عندما نريد العودة مرة أخرى إلى المجلد home directory كل ما علينا فعله استخدام الأمر cd فقط مع العلامة~ ليصبح الأمر بالشكل التالى وبعدها نضغط على الزر Enter :

نفترض مثلا أننا على المسار /var/cache ونتأكد من ذلك أولا :

كود PHP:

```
oem@ubuntu:/var/cache$ pwd
/var/cache
```

بعد ذلك نقوم بكتابة الأمر cd مع العلامة ~ ونضغط على الزر Enter كما يلى :

كود PHP:

```
oem@ubuntu:/var/cache$ cd ~
```

ليكون الناتج كالتالى :

كود PHP:

```
oem@ubuntu:/var/cache$ cd ~
oem@ubuntu:~$
```

وللتأكد نقوم بكتابة الأمر pwd كما يلى :

كود PHP:

```
oem@ubuntu:pwd
/home/oem
```

من ضمن المميزات الرائعة التى يمكن ادراجها مع الأمر cd هى العلامة- أو ال dash والتى تعنى "

"previous directory أى المجلد السابق او الدليل السابق الذى كنا عليه ولفهم ذلك نفترض أننا كنا على المسار var/cache/ وقمنا بالتغيير إلى مسار آخر لأداء مهمة معينة وليكن المسار home/oem/ وبعد أن فرغنا من أداء المهمة أردنا الرجوع إلى المسار السابق ألا وهو var/cache/ فكل ما علينا فعله التالى :

فى البداية كنا على المسار var/cache/ كما بالشكل :

كود PHP:

```
oem@ubuntu:/var/cache$
```

بعد ذلك قمنا بالتغيير إلى المسار home/oem/ كما بالشكل :

كود PHP:

```
oem@ubuntu:/var/cache$ cd /home/oem
```

نقوم بالضغط على زر Enter لنتحول إلى المسار الجديد كما يلى :

كود PHP:

```
oem@ubuntu:/home/oem$
```

الآن نريد الرجوع مرة أخرى إلى المسار var/cache/ دون كتابة المسار والتى تقوم بعمل ذلك هى ال - مع الأمر cd ونضغط على الزر Enter كما يلى :

كود PHP:

```
oem@ubuntu:/home/oem$ cd -
/var/cache
oem@ubuntu:/var/cache$
```

وإن شاء الله سوف أكمل الرد فى هذه المشاركة وعذرا على التأخير نظرا لظروف ألمت بى فى الفترة الماضية أسأل الله أن يفرج همنا وهم المسلمين الموحدين آمين

السلام عليكم ورحمة الله وبركاته

السلام عليكم ورحمة الله وبركاته

بعد انقطاع نعود إليكم لنُكمل ما بدأناه ألا وهى الرحلة الشيقة مع سطر الأوامر

الأمر :touch

يعتبر الأمر touch من الأوامر التى نادراً ما نقوم باستخدامها خلال تعاملنا اليومى مع الطرفية ، ولكن إن شاء الله تعالى من خلال رحلتنا الشيقة فى هذه السلسلة سوف نتعرض له أكثر خلال المسيرة وإظهار الإمكانيات المتقدمة التى يقدمها الأمر لنا .وقد يعتقد البعض أن الوظيفة الرئيسية التى يقوم بها الأمر touch هى تحديث وقت الولوج وتنفيذ المهام على ملف ما بواسطة هذا الأمر ، ولكن سيكون الأعتماد على استخدام هذا الأمر بواسطة الوظائف الفرعية التى يقوم بها والتى هى أهم من الوظيفة الرئيسية !! وسنستعرض ذلك بالتفصيل لاحقا .

لاحظ : تستطيع استخدام الأمر touch على ملف ما وتغيير أوقات التعديل والولوج إذا كنت فقط تمتلك تصريح الكتابة على الملف وإلا فسيفشل الأمر فى تنفيذ المهمة !!

والآن نتطرق إلى الجزء العملى المتعلق بالأمر touch فى أبسط صوره دون إضافة أى خيارات أخرى معه ولنرى كيفية تعديل الوقت والتاريخ على ملف معين ، ونبدأ أولا باستعرض الملفات باستخدام الأمر ls كما يلى :

كود :PHP

```
shuffl@P0w3R:~$ ls -l ~
total 1167060
drwxr-xr-x 7 shuffl shuffl      4096 2007-02-21 19:03 aaaa
-rw------- 1 shuffl shuffl    865301 2007-02-21 18:45 arp.html
-rw-r--r-- 1 shuffl shuffl     31295 2006-11-27 10:07 ar
p.html.zip
```

هنا قمنا أولا باستعراض الملفات الموجودة على المجلد home directory والتى ترمز العلامة~ إليه وكانت النتيجة كما بأعلى والآن نريد مثلا تغيير الوقت والتاريخ للمجلد ونلاحظ أن آخر وقت تم التعديل على الملف كانت الساعة 19:03 والتاريخ هو 2007-02-21 كما يلى :

كود :PHP

```
shuffl@P0w3R:~$ touch aaaa
shuffl@P0w3R:~$ ls -l~
total 1167060
drwxr-xr-x 7 shuffl shuffl      4096 2007-02-22 06:17 aaaa
-rw------- 1 shuffl shuffl    865301 2007-02-21 18:45 arp.html
-rw-r--r-- 1 shuffl shuffl     31295 2006-11-27 10:07 ar
p.html.zip
```

كما ترون تم تحديث الأوقات الخاصة بالمجلد aaaa لتُصبح هى الأوقات الفعلية الحالية لدينا,.ويتضح من ذلك أن الأمر touch يقوم بالتعديل على كل من الملفات والمجلدات على حد سواء بمعنى إذا كان لديك ملفات نصية أو أى نوع ترغب فى تغيير وقت التعديل عليه فما عليك إلا كتابة الأمر مصحوبا بإسم الملف المُراد .

ولكن قد يتبادر إلى الأذهان سؤالاً بخصوص الأمر ، ألا وهو هل الأمر touch يقوم بتغيير وقت التعديل على

الملفات إلى الأوقات الحالية فقط أم أى وقت وتاريخ أريده ؟؟

تعلمنا فيما مضى أن لينوكس عالماً آخر ، ففيه تمتلك القدرة على تطويع أى شيء تريده لأنه نظام مرن بما فيه الكفاية ليفعل لك ما تريده ، ولذلك تستطيع بالطبع عند استخدام الأمر touch لتغيير أوقات التعديل على ملفات تريدها أن تغير أوقات التعديل إلى الأوقات التى تختارها أنت وبالتالى كل ما نحتاجه هو اضافة خيار إلى الأمر touch ليقوم بهذه المهمة وهذا الخيار هو t- والذى عنده إضافته للأمر يقوم بتغيير الأوقات التى تختارها بسلاسة ومرونة رائعة والجدول التالى يوضح الصيغة العامة لتغيير أوقات الملفات كالتالى
: [[CC]YY]MMDDhhmm[.ss]

كود PHP:

```
character                 Meaning

CC                        First two characters of a four-digit
year

YY                        Two-digit year:
                            * if from (00-68) , assumes that fi
rst two
                              digits are 20
                            * if form (69-99) , assumes that fi
rst two
                              digits are 19
                            * if nothing , assumes current year

MM                       Month  (01-12)

DD                       Day    (01-31)

hh                       Hour   (01-23)

mm                       Minute (00-59)

ss                       Second (00-59)
```

والآن قد رأيت الجدول السابق لكيفية وضع القيم الخاصة بالتاريخ والوقت سواء كان التاريخ سنين ، شهور ، أيام وكذلك الوقت ساعات ، دقائق ، ثوان .

والآن إلى ملاحظة مهمة وهى متعلقة بالأرقام التى ليست على صورة ثنائية بمعنى ، إذا كان الرقم لديك مكون من خانة واحدة فقط كرقم 1 أو 2 أو 3 إلخ ينبغى أن تقوم بكتابة صفر قبل الرقم حتى تقوم الطرفية بفهم صيغة تغيير الوقت والتاريخ بشكل سليم . والآن نتطرق إلى بعض الأمثلة المتعلقة بالأمر touch بعد إضافته إلى الخيار t- لكى نقوم بوضع تاريخ ووقت معين من اختيارنا كالتالى :

مثال : قمنا بإنشاء ملف جديد باسم wireless.log والآن نريد استعراض كافة المعلومات الخاصة بالملف بداية من التصاريخ الموضوعة عليه ، واستعراض المالك له والمجموعة كذلك آخر وقت تم التعديل فيه على الملف كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ ls -l wireless.log
-rw-r--r-- 1 shuffl shuffl 0 2007-04-16 18:17 wireless.log
```

هنا فى المثال الوقت الذى تم آخر تعديل فيه على الملف هو الساعة 18:17 والتاريخ هو 2007-04-16 .

نريد الآن تغيير ذلك الوقت والتاريخ لتكون الساعة 00:00 ويكون التاريخ 1992-12-31 ويكون ذلك كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ touch -t 9212310000 wireless.log
shuffl@P0w3R:~/Desktop$ ls -l wireless.log
-rw-r--r-- 1 shuffl shuffl 0 1992-12-31 00:00 wireless.log
```

كما ترون تمت المهمة بنجاح وقمنا بتعديل كلاً من الوقت والتاريخ اللذان قمنا باختيارهما . نأتى إلى مثال آخر ونريد تغيير الوقت إلى الساعة 02:34 والتاريخ هو 2034-05-17 عن طريق عمل التالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ touch -t 3405170234 wireless.log
shuffl@P0w3R:~/Desktop$ ls -l wireless.log
-rw-r--r-- 1 shuffl shuffl 0 2034-05-17 02:34 wireless.log
```

وبالفعل تم لنا ذلك أيضاً !! . ونأتى إلى مثال أخير وهو تغيير الوقت إلى الساعة 17:03 والتاريخ إلى 19-2007-10 أى السنة الحالية :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ touch -t 10191703 wireless.log
shuffl@P0w3R:~/Desktop$ ls -l wireless.log
-rw-r--r-- 1 shuffl shuffl 0 2007-10-19 17:03 wireless.log
```

والآن نشرح كيفية وضع القيم فى الأمثلة السابقة ، فى المثال الأول أردنا تغيير التاريخ إلى 1992-12-31 قمنا أولا بكتابة السنة المُرادة من اليسار وهى 92 ثم الشهر المُراد وهو 12 ثم اليوم المُراد وهو 31 ولكن قد يستوقفنى البعض ويقول لى لما لم تكتب صيغة السنة كاملة من أربعة أرقام لتُصبح 1992 بدلا من 92 أو بصيغة أخرى كيف علمت الطرفية أنّ السنة التى أريدها هى 1992 وليست مثلا 2099 أو شىءٌ من ذاك القبيل ؟؟

لو تتذكرون بأعلى وضعت جدول يحدد صيغ وضع الوقت والتاريخ فالبداية كانت من الحرفين CC واللذان يرمزان إلى أول رقمين من أربعة أرقام من السنة وبعدهما الحرفان YY واللذان يرمزان إلى الرقمين الآخرين من أرقام السنة وذكرت أنّ للحرفين YY ثلاث حالات :

الأولى : أن يكون الرقمان فى المدة من سنة رقمها 00 إلى سنة رقمها 68 وهنا تعتبر الطرفية أن أول رقمين هما 20 وبالتالى يصبح نطاق الأرقام من 00 إلى 68 خاص بالألفية الثانية أو من سنة 2000 إلى سنة 2068 وهذه الحالة تمثل المثال رقم واحد.

الثانية : أن يكون الرقمان فى المدة من سنة رقمها 69 إلى سنة رقمها 99 وهنا تعتبر الطرفية أن أول رقمين هما 19 وبالتالى يصبح نطاق الأرقام من 69 إلى 99 خاص بالقرن التاسع عشر أو من سنة 1969 إلى سنة 1999 وهى الحالة تمثل المثال رقم اثنان.

الثالثة : إذا لم تضع أرقاماً فتعتبر الطرفية أنّ السنة المُرادة هى السنة الحالية وهذه الحالة تمثل المثال رقم ثلاثة.

توضيح : كما ذكرت أعلى أنّ للسنة تكتب أربعة أرقام أى من الممكن أن تكتب سنة 1992 بهذه الشكل أو أن تكتب الرقمين 92 فقط ، كذلك يمكنك كتابة سنة 2022 هكذا أو أن تكتب السنة مُكتفيا بالرقمين 22 فقط .

وبديهى يكون الأمر بالمثل بالنسبة لكل من وضع الشهر ثم اليوم ثم الساعة وهكذا وأظن لا حاجة لشرح هذه الأمور فالأمر قد اتضح .

تعلمنا الآن أن الأمر touch يقوم بمهام رائعة تخص تعديل الوقت والتاريخ ولكن مازال الأمر يحوى الكثير فى جعبته !! نعم ،ولكن حقيقةً المهام السابقة التى قام بها الأمر ليست هى المهام الرئيسية التى يقوم بها الأفراد عادة فالأمر touch له استخدام أكثر إثارة من المهام السابقة وهذا الإستخدام يكمن فى أنه يُعطيك القدرة على إنشاء ملفات فارغة !!

قد يندهش البعض لذلك ولكن تعال معى لترى كيف يمكن عمل ذلك كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ touch test.txt
shuffl@P0w3R:~/Desktop$ ls -l

rw-r--r--   1 shuffl shuffl       0 2007-04-17 09:00 test.txt-
rw-r--r--   1 shuffl shuffl       0 1969-10-19 17:03 wireles-
s.log
rw-r--r--   1 shuffl shuffl     493 2006-10-02 18:08 xcha-
  t.desktop
```

نعم ! لا تندهش فعلا قام الأمر بإنشاء ملف جديد وهو الملف test.txt وبالتالى نستنتج من ذلك أنه يُمكنك إنشاء أى ملف بأى صيغة تُريدها باستخدام الأمر touch و كلُ ما عليك عمله هو مجرد فقط كتابة الأمر متبوعاً بإسم الملف الذى تود إنشاؤه .

الأمر : mkdir

(Create a New Directory)

استعرضنا فى الأمر السابق touch بعض المهام الجيدة التى يقوم بها من تعديل وقت وتواريخ الملفات إلى إمكانية إنشائه ملفات فارغة وبعد أن تعرفنا على هذه المهام نريد التقدم خطوة أخرى وهى كيفية إنشاء مجلدات ؟؟

هنا يأتى دور الأمر mkdir والذى هو اختصار كلمتىّ make directory ، وكل ما عليك عمله لإنشاء مجلد فارغ هو مجرد كتابة الأمر mkdir متبوعا بإسم المجلد بالشكل التالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ mkdir test
shuffl@P0w3R:~/Desktop$ ls -l

drwxr-xr-x   8 shuffl shuffl    4096 2007-03-11 07:56 terminolg
```

```
ies
drwxr-xr-x   2 shuffl shuffl    4096 2007-04-17 09:32 test
-rw-r--r--   1 shuffl shuffl       0 2007-04-17 09:00 test.txt
```

ملحوظة : معظم التوزيعات عن إنشاء مجلدات باستخدام الأمر mkdir يكون التصريح الإفتراضى للمجلد هو القراءة والكتابة والتنفيذ لصاحب الجلد أو المالك ، والقراءة والتنفيذ لكلاً المجموعة وأى مستخدم آخر على التوزيعة وقد بينت هذا الأمر فى موضوع التصاريح على الملفات والمجلدات بأنظمة unix-like فلتراجعة على هذا الرابط :

http://www.linuxac.org/forum/showthread.php?t=2437

والأمر mkdir ذكى بما فيه الكفاية ، بمعنى حينما تود إنشاء ملف يحمل نفس الإسم فى مسار معين على الفور يُخبرك الأمر أنه لا يستطيع تكرار المجلد لأنه موجود مُسبّقاً ولنرى ذلك كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ mkdir test
mkdir: cannot create directory `test': File exists
```

الآن تعلمنا خطوة أخرى وهى إنشاء مجلد فارغ باستخدام أمر جديد وهو الأمر mkdir ، ولكن كما قلنا فى البداية أن لكل أمر من أوامر الطرفية بعض الخيارات التى يتم إضافتها إليه لتُكسب الأمر قوةً ومرونةً حقيقية وهذا ما يحدث للأمر mkdir فَيُوجد لدى الأمر بعض الخيارات المهمة نستعرض البعض منها على سبيل المثال :

mkdir -p

(Create a New Directory and Necessary Subdirectories)

أول خيار نستعرضه مع الأمر mkdir هو الخيار p- والذى يرمز إلى parents-- ، حيث يُمكنك هذا الخيار عند دمجه مع الأمر من إنشاء المجلد الذى توده بالإضافة إلى مجلدات فرعية تحت هذا المجلد دون الحاجة إلى إنشاء كل مجلد فرعى على حده !! دون أن نكثر الكلام نأخذ مثالاً عملياً على ذلك :

مثال : تود إنشاء مجلد مثلا تريد فيه بعض الصور وتود أيضا إنشاء مجلدات فرعية تحت هذا المجلد كل منها يشمل قسم معين ، نستطيع عمل ذلك كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ mkdir -p pictures/personal/natural
shuffl@P0w3R:~/Desktop$ ls -l

drwxr-xr-x   3 shuffl shuffl    4096 2007-04-17 09:48 pictures
drwxr-xr-x   8 shuffl shuffl    4096 2007-03-11 07:56 terminolg
ies
-rw-r--r--   1 shuffl shuffl       0 2007-04-17 09:00 test.txt


shuffl@P0w3R:~/Desktop$ cd pictures/
```

```
shuffl@P0w3R:~/Desktop/pictures$ ls -l

drwxr-xr-x 3 shuffl shuffl 4096 2007-04-17 09:48 personal


shuffl@P0w3R:~/Desktop/pictures$ cd personal/
shuffl@P0w3R:~/Desktop/pictures/personal$ ls -l

drwxr-xr-x 2 shuffl shuffl 4096 2007-04-17 09:48 natural
```

كما ترى قام الأمر باستخدام الخيار p- بإنشاء المجلد الرئيسى وهو pictures ، ثم قام بإنشاء مجلد فرعى تحت المجلد pictures وهو المجلد personal ، ثم قام أخيرا بإنشاء مجلد فرعى آخر تحت المجلد الفرعى personal وهو المجلد natural .

mkdir -v

(Find Out What mkdir Is Doing As It Acts)

الخيار الذى سنتحدث عنه الآن مُفيد فى بعض الحالات أثناء قيامك أداء مهمة معينة باستخدام أمر مُعين يمكن إضافة هذا الخيار معه ، بمعنى حينما قمنا فى النقطة السابقة والتى تحدثنا فيها عن الخيار p- لم نعلم كيف بدأ الأمر mkdir فى إنشاء المجلدات ، هل بدأ أولا بإنشاء المجلد الرئيسى ، أم بدأ أولا بإنشاء آخر مجلد فرعى وهكذا ، ولذلك الخيار v- يتيح لنا إمكانية معرفة أى المجلدات تم البدء فى إنشائها أولا والخيار v- يرمز إلى verbose-- وبالتالى نستطيع تكرار المثل السابق ولكن الجديد هنا إضافة خياران فى نفس واحد هما الخيار p- والخيار v- كالتالى :

كود PHP:

```
shuffl@P0w3R:~/Desktop$ mkdir -vp picutres/personal/natural

mkdir: created directory `picutres'
mkdir: created directory `picutres/personal'
mkdir: created directory `picutres/personal/natural'
```

كما تلاحظ قام الأمر mkdir أولا بإنشاء المجلد pictures ثم قام بعد ذلك بإنشاء المجلد personal داخل المجلد pictures ثم قام أخيراً بإنشاء المجلد natural داخل المجلد personal والذى هو بدوره داخل المجلد natural .

ملحوظة : لا يفضل استخدام الخيار v- مع أوامر فك الضغط عن الملفات حيث يقوم الخيار بمضاعفة الوقت اللازم لعملية الفك .

إلى هذا الحد نكتفى فى هذه المشاركة ونُكمل فى المشاركات القادمة بإذنه تعالى وما أسألكم هو الدعاء لى بظاهر الغيب فقط .

وأتمنى من الإخوة عدم الرد فى هذا الموضوع حيث أنى قمت بالتنبيه فى أول الموضوع أن الردود والإستفسارات يتم وضعها فى هذا الموضوع :

http://www.linuxac.org/forum/showthread.php?t=1603

دمتم بحفظ الله وعنايته

السلام عليكم ورحمة الله وبركاته

السلام عليكم ورحمة الله وبركاته

بعد فترة طويلة نعود إليكم من جديد ، وها نحن نواصل المسيرة الشيقة مع طرفية جنو/لينوكس مرة أخرى نتمنى لكم كامل المتعة والإستفادة فى هذه الرحلة .

الأمر cp

عندما نتحدث عن عملية النسخ فإننا نتحدث عن عملية قد تكون شبه يومية بالنسبة للبعض ، بل قد نستطيع القول بأنها عملية شبه وقتيه تحدث بين الفينة والأخرى فى اليوم الواحد ، لذا عندما نتعامل مع الطرفية استوجب الأمر وجود أمر ما يسد النقص فى تلك الجهة ويكون هو المسئول الأول عن تلك العملية .

تلك كانت مقدمة بسيطة للأمر cp والذى سوف أتناوله إن شاء الله فى معرض حديثى ، لكن فى البداية لنبدأ بشكل مُبسط لكيفية إستخدام الأمر cp، كل ما عليك فعله هو كتابة الأمر cp متبوعاً بإسم الملف الذى تود نسخه ثم أخيراً متبوعا بإسم الملف الجديد المنسوخ ! عملية سهلة أليس كذلك ؟؟ فكر فى العناصر المكونة للأمر cp وهى فى البداية العنصر الرئيسى لعملية النسخ وهو الأمر ذاته ثم بعد ذلك الملف الذى تود نسخه من مصدر معين ثم أخيراً العنصر الثالث وهو الملف المنسوخ الذى تود الحصول عليه ، لذا نستطيع التعبير أيضاً بطريقة أخرى عن العناصر الرئيسية المكونة لعملية النسخ فى صورة أبسط من ذلك بكثير كالتالى :

كود PHP:

```
cp "source المصدر الذى نقوم بالنسخ منه" "Destination الجهة التى نقوم بالنسخ إليها"
```

ولكى نستطيع فهم ما قلناه من كلمات نظرية لابد من تسخين أيدينا قليلا ولذا نسرع بفتح محاكى الطرفية ونقوم بتنفيذ التالى :

مثال 1 :

كود PHP:

```
[1]eng1n33r@p0w3r:~$ pwd
/export/home/eng1n33r
```

كود PHP:

```
[2]eng1n33r@p0w3r:~$ ls
debian.vdi
language
win.vdi
```

كود PHP:

```
[3]eng1n33r@p0w3r:~$ cp language language_bakup
```

كود PHP:

```
eng1n33r@p0w3r:~$ ls
debian.vdi
```

```
language
language_bakup
win.vdi
```

الآن نتطرق إلى شرح المثال :

1- فى البداية قمنا بطباعة الدليل الحالى الذى نعمل عليه بإستخدام الأمر pwd (لمعرفة المزيد عن الأمر راجع الموضوع أعلى).

2- بعد ذلك قمنا بإستعراض الملفات التى توجد داخل ذلك المجلد بإستخدام الأمر ls (لمعرفة المزيد عن الأمر راجع الموضوع أعلى).

3- قمنا بعملية النسخ فى أبسط صورها التى كان الغرض منها هو عمل نسخة من الملف "language" بإستخدام الأمر cp وكان صيغة الأمر فى المثال هو إسم الملف الأصلى الذى نود نسخه وهو "language" متبوعا بإسم الملف الجديد المنسوخ والذى كان فى المثال "language_bakup" .

4- أخيراً قمنا بإستعراض الملفات داخل المجلد مرة أخرى بإستخدام الأمر ls للتأكيد فعليا من وجود نسخة من الملف كما فعلنا .

كما ترى فى هذا المثال أن عملية النسخ بسيطة جدا كل ما أردنا فعله هو مجرد نسخ ملف مرة أخرى داخل نفس المجلد الذى يوجد ، لذا تفاديا لعملية التكرار قمنا بتغيير إسم الملف الجديد المنسوخ إلى إسم آخر .

لكن السؤال الذى قد يدور فى ذهن البعض الآن " لكنى لا أريد نسخ الملف مرتين داخل مجلد واحد ، ما أريد فعله هو نسخ ملف معين إلى مسار آخر هل يمكننى فعل ذلك ؟؟

الإجابة بالطبع تستطيع ، وإلا ما الفائدة إذا من عملية النسخ ، ولكى تلمس ذلك عن قرب نأخد المثال التالى :

مثال 2 :

كود PHP:

```
¹eng1n33r@p0w3r:~$ pwd
/export/home/eng1n33r
```

كود PHP:

```
²eng1n33r@p0w3r:~$ cp language /export/home/eng1n33r/Desktop/
```

كود PHP:

```
³eng1n33r@p0w3r:~$ ls /export/home/eng1n33r/Desktop/
cool-stack topics
LAC_magazine_Jul_Aug_2008.pdf
language
links.txt
```

شرح المثال :

1- فى البداية قمنا بإستخدام الأمر pwd لمعرفة الدليل الحالى الذى نعمل عليه .

2- بعد ذلك قمنا بنسخ الملف "language" من المجلد الحالى الذى نعمل بداخله إلى المسار الجديد "الجهة" التى نود نسخ الملف إليها والتى كانت فى المثال "/export/home/eng1n33r/Desktop/"

ملحوظة : مسارات المجلدات قد تختلف بشكل كلى أو جزئى عما لديك فتنبه لذلك .

3- قمنا أخيرا باستخدام الأمر ls لإستعراض المجلد Desktop للتأكد فعلا من تمام عملية نسخ الملف " language" ، وبالفعل تمت عملية النسخ بنجاح .

إذا نستطيع إستنتاج قاعدة مهمة فى التعامل مع الأمر cp :

عند البدء فى عملية النسخ ينبغى معرفة المصدر "source" الذى تود النسخ ثم بعد ذلك يكون متبوعا بالجهة "destination" المراد النسخ إليه ، هذا كل ما فى الأمر .

تلميح : فى بعض الإحيان قد نريد نسخ بعض الملفات من مسار معين إلى الدليل الحالى الذى نعمل عليه وبالتالى نقوم بكتابة المصدر الذى نود النسخ منه كاملا ونكتفى فقط بالجهة التى نود النسخ إليها وهو الدليل الحالى "current directory" والذى نستطيع التعبير عنه ب "." أو نقطة "dot" .

مثال على ذلك :

كود :PHP

```
eng1n33r@p0w3r:~$ pwd
/export/home/eng1n33r
```

كود :PHP

```
eng1n33r@p0w3r:~$ cp /export/home/eng1n33r/Desktop/Q_plus_A.txt
.
```

إستخدام الأمر cp مع ال WildCards

سابقا قمنا بتوضيح ماهية ال WildCards وكيفية استخدامها فى تسهيل كثير من الأمور لذا إذا كنت لا تعرف ما أعنيه بال WildCards فلتراجع بداية الموضوع فى فقرة " أساسيات لابد من الإلمام بها "

الآن حان وقت الكسل ، فمع دمج ال WildCards مع الأمر cp ستشعر بأن الأمور أصبحت أسهل بكثير من ذى قبل ، فلتتخيل معى هذه العملية فى البداية كان الأمر طبيعى جدا أن تقوم بنسخ ملف واحد ولم تكن لتوجد مشكلة فى ذلك الأمر لكن هل فكرت قليلا أنك فى يوم من الأيام لن تكتفى بنسخ ملف واحد وستحتاج إلى نسخ عدة ملفات ؟؟

أرى أنه قد وصل إليك ما أريد إيصاله ، ولسان حالك بالطبع هل سأقوم بنسخ الملفات التى أريدها واحدا تلو الآخر ؟؟ مهمة شاقة أليس كذلك ؟

نعم مهمة شاقة أى شاقة ، فلنتفرض أنك تريد نسخ ألف ملف مرة واحدة من مسار أو مجلد إلى آخر هل حينها ستقوم بعمل ذلك ألف مرة ؟؟ أشك لذا يأتى هنا دور "*" أو ال "asterisk" والتى تعنى عند استخدامها "الكل" فمثلا أنا أريد نقل عشرة ملفات من مجلد إلى آخر مرة واحدة فنستطيع التعبير عن العشرة ملفات دفعة واحدة بإستخدام "*" ولكى تتضح الأمور بشكل أكبر لديك نأخذ مثالا .

مثال 3 :

كود PHP:

```
¹eng1n33r@p0w3r:~$ cd Desktop/VOIP/
```

كود PHP:

```
²eng1n33r@p0w3r:~/Desktop/VOIP$ ls
1008.mp3      index.htm        VOIP-Overview.pdf    VOIP-SER-
Part2.pdf
index_files   VOIPfull.pdf   VOIP-SER-Part1.pdf   VOIP-Testbed.pdf
```

كود PHP:

```
³eng1n33r@p0w3r:~/Desktop/VOIP$ cp * /export/home/eng1n33r/Desk
top/akl/
```

كود PHP:

```
eng1n33r@p0w3r:~/Desktop/VOIP$ ls /export/home/eng1n33r/Desktop
/akl/
1008.mp3    VOIPfull.pdf         VOIP-SER-Part1.pdf   VOIP-
Testbed.pdf
index.htm   VOIP-Overview.pdf   VOIP-SER-Part2.pdf
```

شرح المثال :

1- فى البداية قمنا بالولوج إلى الدليل الذى توجد بداخله الملفات التى نريد نسخها بإستخدام الأمر cd .

2- بعد ذلك قمنا بإستعراض الملفات التى توجد داخل ذلك المجلد بإستخدام الأمر ls .

3- الآن قمنا بإستخدام الأمر cp مع * للتعبير أننا نريد نسخ جميع الملفات داخل المجلد VOIP ونسخ هذه الملفات إلى المسار المذكور فى المثال .

4- أخيراً نقوم بالتأكد فعليا من نسخ الملفات إلى الجهة التى اردنا نسخ الملفات إليها بإستخدام الأمر ls .

ونكمل إن شاء الله لاحقاً

السلام عليكم ورحمة الله وبركاته

السلام عليكم ورحمة الله وبركاته

سنتناول اليوم جزئية" كيفية البحث عن ملفات أو مجلدات على نظام التشغيل " ، وسنتطرق إلى أمر مهم يقوم بتلك العملية وهو locate .

مقدمة

الكل يعلم أن الأقراص الصلبة فى تطور مستمر سواء من حيث سرعات القراءة الخاصة بها أو من حيث الأحجام ، وآخر رقم سمعته من حيث السعة بالجيجا بايت هو من شركة Hitachi وكان القرص الصلب بسعة 1024 جيجا بايت أو فيما يعادل واحد تيرا بايت ، لذلك تخيل معى شكل ذلك القرص الصلب بمحتوياته وتخيل معى أيضاً إذا كانت تلك المحتويات غير منظمة ولا تستطيع الوصول إلى ملف معين تريده فما الحل إذاً ؟

هنا تظهر أهمية البحث بشكل عام ، فمن خلال خاصية البحث على نظام التشغيل الذى تعمل عليه سواء كان جنو /لينوكس أو يونكس أو أى نظام تشغيل آخر تستطيع توفير الكثير من الوقت وتوفير معاناة أنت فى غنى عنها لمجرد البحث عن ملف لا تتذكر أين قمت بوضعه ، لكن ما سنتحدث عنه خلال السطور القادمة إن شاء الله بخصوص أدوات البحث تلك ستكون من خلال نظام التشغيل جنو /لينوكس ولا يهمنا غيره ، لكن تجدر الإشارة هنا أن الأمرين locate والأمر find يختلف كلاهما فى فكرة العمل وطريقة البحث وهذا ما سوف نوضحه خلال السطور القادمة ونبدأ تلك الأوامر مع الأمر locate.

الأمر locate

تعلم إسم الملف أو جزء من ذلك الإسم لكن لا تستطيع تحديد مكان ذلك الملف على نظام التشغيل لديك ؟؟ هنا يأتى دور الأمر locate ، فالأمر locate يقوم بالبحث عن أسماء الملفات أو المجلدات أو البرامج أو أجزاء من تلك الإسماء إعتماداً على قاعدة بيانات تكون خاصة به وتكون نتيجة البحث متضمنة كل الأسماء التى تحتوى كلمة البحث أو جزء منها داخل قاعدة البيانات تلك ، بمعنى أن حدود الأمر locate فى عملية البحث الخاصة به لن تتجاوز قاعدة البيانات تلك ، ولذا ينبغى كل فترة تحديث قاعدة البيانات لإدارج المزيد من نتائج البحث ، ولاحقاً سنتطرق إلى كيفية تنفيذ عملية التحديث تلك فلا تقلق من ذلك .

نبدأ الآن فى إستخدام الأمر locate فى أبسط صوره وهى كتابة الأمر متبوعا بكلمة البحث التى تريد البحث عنها بالشكل التالى :

كود PHP:

```
muhammad@debian:~$ locate jigdo

 /root/.jigdo-lite
 /usr/bin/jigdo-file
 /usr/bin/jigdo-lite
 /usr/bin/jigdo-mirror
 /usr/share/doc-base/jigdo-mini-howto
 /usr/share/doc/jigdo-file
 /usr/share/doc/jigdo-file/changelog.Debian.gz
```

```
                .
                .
                .
                .
/var/lib/dpkg/info/jigdo-file.list
/var/lib/dpkg/info/jigdo-file.md5sums
```

كما تلاحظ قمنا فى البداية بكتابة الأمر locate ، ثم بعد ذلك قمنا بإتباعه بكملة البحث التى تريد البحث عنها والتى كانت jigdo وأخيراً كانت نتيجة الأمر بالشكل السابق وهى كل الملفات التى احتوت على الكلمة jigdo ، لكن ألم تلاحظ شيئاً آخر ؟؟

قد تبادر بالسؤال وتوجهه لى بصيغة أخرى أى شىء ؟؟

لو لاحظت حينما قمنا بكتابة الأمر locate متبوعاً بكلمة البحث jigdo لم تستغرق العملية ثوانى معدودة بل أقل من ذلك وذلك لأننا كما ذكرنا قبل ذلك أن الأمر locate لا يقوم بالبحث عن الكلمات التى تمررها له على نظام التشغيل ككل ، ولكن يقوم بالبحث داخل قاعدة البيانات الخاصة به ، لذا تكون عملية البحث أسرع بكثير من عمليات البحث العادية والتى تكون على نظام التشغيل ككل أو جزء من نظام التشغيل ، لكن هذه الخاصية أى خاصية البحث داخل قاعدة البيانات قد تكون ميزة أو عيب فى نفس الوقت ، بمعنى تكون تلك الخاصية ميزة فى السرعة التى تحصل عليها من أداء عملية البحث وظهور النتائج بشكل أسرع من مثيلاتها ، وقد تكون عيب فى حالة عدم توافر نتائج لعملية بحث معينة تريدها لا تكون داخل قاعدة البيانات الخاصة بالأمر locate ، لذلك البحث داخل قاعدة بيانات منشأة مسبقا سيوفر المزيد من الوقت بشكل كبير ، وقد لا تستغرق عملية البحث عدة لحظات لا ثوانى .

عملية البحث السابقة كانت تتم بشكل عادى دون حدوث أية مشاكل ، لكن ماذا لو أردنا نحن رؤية تلك المشاكل ؟؟

قد يقول البعض هل أنت من هواة المشاكل ؟؟ نعم !! أنا أحب المشاكل بشكل قد لا تتخيله لأنها تساعدنى على تعلم الكثير والكثير وقد لا تصدق أننى أحيانا أجبر نفسى على الوقوع فى مشاكل لا أفقه عنها شيئا لمجرد تعلم بعض الأمور الجديدة !! (أعلم أن البعض قد يتهمنى بالجنون لكن لا يهمنى رأيهم ! P:)

إذاً لسان حالك يقول الآن : ماذا تقصد بتلك المشاكل ؟؟ أو ماهو الداعى لخلق مشاكل قد نكون فى غنى عنها ؟؟

حينما نتعرف على الأمر slocate ستعرف ماهو قصدى بخلق المشاكل أو ما الذى دعانى إلى فعل ذلك ، لكن قبل أن نتعرف أكثر على slocate ألمح بعض التساؤلات فى ذهنك بنظرة غاضبة تندفع متسائلاً " هل هناك فرق بين locate و slocate ؟ "

قبل أن أجيب على ذلك التساؤل قم بتحميل حزمة الأمر slocate إذا كنت أحد مستخدمى توزيعة Debian Gnu/Linux بالشكل التالى :

كود PHP:

```
debian:~# apt-get install slocate
```

الآن قم بإجراء عملية البحث السابقة عن الملف slocate.db كما ذكرنا والتى كانت بالشكل التالى :

كود PHP:

```
muhammad@debian:~$ locate slocate.db
```

الآن ماهى نتيجة عملية البحث التى تظهر لديك فى محاكى الطرفية هل هى مماثلة للنتيجة السابقة أم ماذا ؟؟ أشك فى ذلك !!

بالطبع لم تظهر أى نتيجة لعملية البحث ، ذلك أن المستخدم "muhammad" ليس له صلاحية عرض الملف السابق ألا وهو slocate.db ، وسيقوم محاكى الطرفية بالرجوع مرة أخرى إلى وضع الإستعداد لتلقى أمر آخر .

بالتأكيد الآن تريد معرفة عدم ظهور نتيجة لعملية البحث مع العلم أنك قمت بكتابة الأمر locate بدون أى خطأ إملائى أليس كذلك ؟؟

للإجابة على ذلك التساؤل سنقوم أولاً ببسط بعض الأمور التى تخص الأمر slocate ، الأمر slocate يرمز إلى secure locate ، وتكمن أهمية ذلك الأمر فى الفائدة التى تحصل عليها إذا كنت مدير أحد أنظمة تشغيل جنو/لينوكس فى منع بعض المستخدمين من البحث داخل مجلدات وملفات تخص مستخدمين آخرين على النظام ما لم يتم السماح لهم بذلك ، فمثلا يريد المستخدم " muhammad" البحث عن الملف السابق slocate.db وذلك الملف قد يكون خاص بالمستخدم "root" فحينها لن يتمكن المستخدم " muhammad" من رؤية نتيجة لعملية البحث وظهور نتائج لتلك العملية مالم يتم منح التصاريح الخاصة على ذلك الملف للقيام بذلك .

الآن لكى نتمكن من إجراء تلك عملية البحث بشكل طبيعى سنقوم بإستخدام حساب المستخدم root عن طريق الأمر su - :

كود PHP:

```
muhammad@debian:~$ su -
Password:
debian:~#
```

الآن قم بكتابة الأمر locate مرة أخرى متبوعاً بإسم الملف slocate.db كما يلى :

كود PHP:

```
debian:~# locate slocate.db
 /var/lib/slocate/slocate.db
```

كما تلاحظ ظهور نتيجة لعملية البحث بعد إستخدام حساب المستخدم "root" ، أيضا هناك مسألة مهمة أخرى ينبغى الإشارة إليها بخصوص الأمر slocate ألا وهى :

عند وجود كلا الأمرين الأمر locate والأمر slocate على نفس التوزيعة فحينها أسبقية التنفيذ ستكون للأمر slocate أولاً ثم بعد ذلك سيتم تنفيذ الأمر locate ولكى تتحق من ذلك قم بتنفيذ الأمر التالى :

كود PHP:

```
muhammad@debian:~$ ls -l /usr/bin/locate

 root root  /usr/bin/locate -> slocate
```

ملحوظة : النتيجة السابقة ناتجة من عمل soft link للأمر locate على المسار /usr/bin/locate مع الأمر slocate والعلامة <- تدل على ذلك .

بعد أن استعرضنا بعض المفاهيم الأساسية التى تخص الأمر locate وكيفية تنفيذ عملية البحث من خلاله سنتطرق الآن إلى بعض الخيارات التى يمكن أن تأتى مع ذلك الأمر لتُكسبه بعض الميزات الإضافية وعمل مهام أكثر .

locate -i

خلال السطور السابقة قمنا بإستخدام الأمر locate لإجراء عملية البحث ولكن لم نكن لنهتم كثير هل إسم الملف يوجد به حروف صغيرة أى lower case أو حروف كبيرة أى Upper case ، لذلك بدمج الخيار -i مع الأمر locate ستجتنب تلك المشكلة تماماً لأنه كما تعلم أن نظام التشغيل جنو/لينوكس حساس تجاه الحروف الكبيرة والصغيرة فالكلمة mysql تختلف عن MySQL تختلف عن MySql وهكذا ، ولذا عند وضع -i مع locate لن تحدث مشاكل بسبب تلك العملية ونأخذ مثالاً عملياً على ذلك :

كود PHP:

```
muhammad@debian:~$ locate -i mysql.txt

/home/muhammad/MysqL.txt
/home/muhammad/mysql.txt
/home/muhammad/MySQL.txt
```

بالفعل تم إهمال حساسية الحروف خلال عملية البحث وظهرت النتيجة لكل الملفات التى تحت إسم mysql.txt على إختلاف حالة الأحرف فيها .

locate -n

الملف السابق الذى قمنا ببناء عملية البحث عليه كان الملف mysql.txt ، وكما رأينا أن نتيجة البحث تضمنت ثلاث نتائج فقط وبالتالى لم يكن الأمر صعباً فى تحديد أين من الملفات الثلاث نريد أو على أي المسارات توجد ، لكن ماذا لو كانت نتيجة البحث كبيرة بالشكل الذى قد يجعلها آلاف النتائج لملف واحد هل ستستطيع حينها معرفة مرادك من خلال نتائج البحث ؟ أظن أن الأمر سيكون صعبا لذا يوفر الخيار -n عند دمجه مع الأمر locate إمكانية تحديد عدد نتائج عملية البحث التى سوف تظهر ولكى يصل إليك ما أود قوله نأخذ المثال التالى :

مثال : تريد أول ثلاث نتائج لعملية البحث عن الملف jigdo كيف ستكون صيغة الأمر locate مع الخيار -n ؟؟

كود PHP:

```
muhammad@debian:~$ locate -n 3 jigdo

 /var/lib/dpkg/info/jigdo-file.postinst
 /var/lib/dpkg/info/jigdo-file.list
 /var/lib/dpkg/info/jigdo-file.md5sums
```

كما تلاحظ قمنا بكتابة الأمر locate أولاً ثم بعد ذلك أتبعناه بالخيار -n ثم بعد ذلك قمنا بتحديد العدد لنتائج عملية البحث ثم أخيراً إسم الملف الذى نود البحث عنه ، وبالطبع تستطيع تغيير الرقم 3 إلى أى رقم تريده للنتائج الخارجة من عملية البحث .

الأمر updatedb

كنا قد ذكرنا فى بداية حديثنا عن الأمر locate ميزة السرعة فى جلب نتائج عملية البحث دون أن تستغرق عملية البحث عدة ثوانى على الأكثر وذلك نتيجة إعتماد الأمر على قاعدة بيانات خاصة به تضم أسماء الملف على نظام التشغيل ، لكن فى بعض الأحيان عندما نقوم بتنفيذ الأمر locate لا تظهر نتائج لعملية البحث وقد يكون ذلك الأمر راجعاً إما لسببين :

الأمر الأول : هو عدم وجود الملف الذى يتم البحث عنه على نظام التشغيل .

الأمر الثانى : هو أن قاعدة البيانات الخاصة بالأمر لم يحدث لها تحديث بعد .

لذا دور الأمر updatedb هنا هو تحديث قاعدة البيانات التى يعتمد عليها الأمر locate بشكل يدوى ، لكن ينبغى الإشارة إلى أمر مهم أن الأمر locate يقوم بتحديث تلك القاعدة بشكل يومى لكن الغرض من تحديث القاعدة بشكل يدوى هو العثور على ملفات قد قمنا بإضافتها قبل بدء عملية التحديث التى تحدث بشكل تلقائى .

الآن نريد تنفيذ عملية تحديث قاعدة البيانات التى يعتمد عليها الأمر locate بشكل يدوى كيف لنا فعل ذلك ؟؟

الأمر بسيط جدا قم بالدخول بحساب المستخدم root ونفذ الأمر على صورته البسيطة هكذا :

كود PHP:

```
debian:~# updatedb
```

بعد ذلك قم بالضغط على زر Enter وانتظر قليلا حتى يفرغ الأمر من تحديث قاعدة البيانات .

ملحوظة : تنفيذ عملية تحديث قاعدة البيانات بشكل يدوى بإستخدام الأمر updatedb قد يستغرق بعض الوقت وذلك إعتمادا على كمية الملفات والمجلدات التى تملكها ، كذلك سرعة جهاز الحاسوب والعتاد الخاص به فكلما كانت سرعة الجهاز أعلى كلما تمت عملية التحديث فى وقت أقل .

تلميح : تستطيع معرفة الوقت الذى استغرقته عملية التحديث من خلال الأمر time وتكون الصيغة للأمرين معاً بالشكل التالى :

كود PHP:

```
debian:~# time updatedb
```

وتكون النتيجة بالشكل التالى :

كود PHP:

```
debian:~# time updatedb

real     1m14.912s
user     0m0.144s
sys      0m0.428s
```

الآن قد يتساءل البعض ماهو المسار الذى يحتوى قاعدة البيانات الخاصة بالأمر locate وكيف لى بعرض محتويات تلك القاعدة ؟؟

قاعدة البيانات التى تخص الأمر locate عبارة عن binary database وليست فى صورة نصية يمكن تصفح محتوياتها ، لكن بما أن تلك القاعدة تحتوى على متغيرات نصية لأسماء الملفات على نظام التشغيل فلذا يمكن تصفح تلك المتغيرات النصية بإستخدام الأمر strings بالشكل التالى :

كود PHP:

```
debian:~# strings /var/lib/slocate/slocate.db | less
```

طبعا توجد الكثير من الخيارات التى تأتى مدمجة مع الأمر locate وتستطيع معرفة المزيد بإستخدام الأمر man بالشكل التالى :

كود PHP:

```
debian:~# man locate
```

ملحوظة : هذه المشاركة كنت قد طرحتها فى موضوع مستقل لكن رأيت وضعها هنا للمحافظة على تسلسل الموضوع ووضع كل الأوامر فى موضوع واحد

السلام عليكم ورحمة الله وبركاته

محور حديثنا فى هذه الجزئية سيكون عن كلا من الأمر history والأمر alias وكلاهما يمثل أهمية لمستخدمى جنو/لينوكس على مدار العمل اليومى وسنبدأ رحلتنا مع الأمر history .

الأمر history :

هل فكرت يوما فى أن ما تكتبه من أوامر بشكل اعتيادى يتم تسجيله دون أن تعطى لذلك بالا ؟؟ قد يندهش البعض ويتساءل هل أنا مراقب ؟

مممم كلا الأمر ليس كذلك لكن عندما تقوم بكتابة أمر وتمريره إلى الطرفية فيوجد هناك من يقوم بحفظ كل ما تكتبه سواء كان بشكل صحيح أم

بشكل خاطىء فى ملف خارجى يدعى bash_history. ، هذا الملف يقوم بحفظ الأوامر التى تكتبها بترتيب زمنى تصاعدى أى بداية من أول أمر

قمت بتمريره إلى الطرفية انتهاء بآخر أمر قمت بكتابته وهكذا

الملف bash_history. يوجد داخل ملجد ال home directory الخاص بك ، لكن بشكل طبيعى هذا الملف لا تستطيع ملاحظته لأنه يتم أخفاؤه بشكل افتراضى (لاحظ أن اسم الملف يبدأ ب dot أو نقطة) ، وبشكل افتراضى أيضا يقوم ذلك الملف بتخزين آخر خمسمائة أمر (500) بداخله ويتم حذف الأوامر تباعا الأقدم فالأقدم ليبقى عدد الأوامر المحفوظة داخل الملف 500 أمر .

والآن إذا أردت استخدام الأمر لمعرفة ما يفعل قم بتنفيذ التالى :

كود PHP:

```
muhammad@ubuntu:~$ history

   452   cat /boot/grub/menu.lst | grep ubuntu
   453   cat /boot/grub/menu.lst
   454   man chroot
   455   man compiz
   456   history
     :
     :
     :
     :
```

ماذا تلاحظ بعد كتابة الأمر ؟ قد تندهش بالفعل قام الأمر بسرد مجمل الأوامر التى قمت بادخالها مؤخرا ويعتمد ذلك على مدى استعمالك لمحاكى الطرفية ، بمعنى آخر هل تستخدم الطرفية بشكل كبير أو بين الحين والآخر وهكذا .....

فمثلا فى حالتى حتى الآن قمت بتمرير 456 أمر إلى الطرفية ونظرا لكثرة الأوامر فى حالتى فلم يظهر

منها أمامى على الشاشة إلا بعض الأوامر التى قمت بكتابتها مؤخرا ، لكن إذا اردت معرفة مجموعة الأوامر الأولى التى قمت بكتابتها فقم بدمج الأمر history مع الأمر less بالشكل التالى :

كود PHP:

```
  muhammad@ubuntu:~$ history | less
```

ليكون ناتج دمج الأمرين بالشكل التالى :

كود PHP:

```
    1  sudo apt-cache search compiz
    2  sudo apt-get install compiz-gnome
    3  sudo apt-get install compiz
    4  sudo apt-get install compizconfig-settings-manager
    5  sudo nano /etc/apt/sources.list
    6  clear
    7  sudo apt-get update
    8  sudo apt-get install compizconfig-settings-manager
    :
    :
    :
```

وبالتالى تستطيع تصفح الأوامر بشكل مرتب باستخدام الزر Enter لمعرفة كل الأوامر التى قمت بادخالها مؤخرا

تنبيه : الآن قد علمتم الحكمة من عدم رؤية كلمة المرور التى تقوم بكتابتها باستمرار ، وبالتالى تفوت الفرصة على العابثين بملفاتك والمعلومات التى تخصك .

خيارات الأمر history

قد يتساءل البعض هل المعلومات السابقة هى كل ما تخص الأمر history ؟؟ قطعا لا "!!" مازال الأمر يحتفظ ببعض الخيارات التى تُكسبه قوة إضافية .

الخاصية الأولى : العلامتان "!!" والتى تعنى تنفيذ آخر أمر قمت بإدخاله ، فمثلا آخر أمر قمت بادخاله هو الأمر pwd لذا عندما تقوم كتابة الخاصية !! فيستم تنفيذ الأمر مرة أخرى ولاحظ معى التالى :

فى البداية سأقوم بكتابة الأمر pwd لتصفح محتويات مجلد ما بالشكل التالى :

كود PHP:

```
muhammad@ubuntu:~$ pwd
/home/muhammad
```

و عند كتابة العلامتان "!!" فسنحصل على نفس النتيجة السابقة بالشكل التالى :

كود PHP:

```
muhammad@ubuntu:~$ !!
pwd
/home/muhammad
```

ماذا تستنتج ؟ التفسير البديهى لذلك الأمر هو أن العلامتان "!!" تقومان بالبحث عن آخر أمر داخل الملف .bash_history ثم بعد ذلك تقوم بتنفيذ ذلك الأمر مرة أخرى ، ويؤكد كلامنا السابق أن الأمر pwd تم كتابته أولا ثم بعد ذلك تم تنفيذه مرة .

الخاصية الثانية : "![##]" (تعنى تنفيذ أمر سابق برقمه)

من خلال هذه الخاصية نستطيع تنفيذ أمر سابق برقمه الخاص به بمعنى ، من خلال تنفيذنا للأمر history فى السابق رأينا أن كل أمر يتم تنفيذ يأخذ رقما محددا داخل الملف .bash_history وهكذا ، لذا لكى نستطيع تنفيذ رقم بعينه سيكون ذلك من خلال استخدام "![##]" ولنأخذ مثالا على ذلك :

كود PHP:

```
muhammad@ubuntu:~$ history
  :
  :
 475  clear
 476  ls
 477  pwd
  :
  :
```

بعد ذلك نستخدم العلامة "!" مع رقم الأمر بالشكل التالى :

كود PHP:

```
muhammad@ubuntu:~$ !477
pwd
/home/muhammad
```

كما تلاحظ فى المثال السابق قمنا أولاً باستخدام الأمر history لمعرفة أرقام بعض الأوامر التى قمنا بكتابتها ، بعد ذلك قمت باختيار رقم الأمر pwd وهو 477 ، وأخيرا قمت بتنفيذ الأمر مرة أخرى برقمه باستخدام العلامة "!" .

الخاصية الثالثة : "![string]" (تعنى تنفيذ أمر بإستخدام بعض حروفه)

القدرة على تنفيذ أمر مرة أخرى باستخدام الرقم المناظر له أمر جيد ، لكن هل ستسعفك الذاكرة دوما لتذكر تلك الأوامر ؟ أظن لا لذا لابد من حل بديل ألا وهو استخدام العلامة "!" مع بعض الحروف من الأمر ولنأخذ مثالا على ذلك :

كود :PHP

```
muhammad@ubuntu:~$ cat /home/muhammad/mysql

you can use them as always
try another one
will that help ?
```

كود :PHP

```
muhammad@ubuntu:~$ cd Desktop/
muhammad@ubuntu:~/Desktop$  !ca
```

كود :PHP

```
cat /home/muhammad/mysql
you can use them as always
try another one
will that help ?
```

شرح المثال : فى البداية قمت بتصفح محتويات ملف نصى باسم mysql باستخدام الأمر cat بعد ذلك قمت بالولوج إلى مجلد ال Desktop لمجرد تغيير الأوامر ليس إلا ثم أخيرا قمت باستخدام العلامة "!" مع بعض الحروف من الأمر cat وهما الحرفان ca فكانت النتيجة نفسها كما تلاحظ .

الخاصية الرابعة : Ctrl+r

هذه الخاصية تعتبر أهم الخصائص على الإطلاق لنظراً للفائدة التى تعود علينا من استخدامنا لها ، وليتح استخدام الخاصية قم على الفور بفتح محاكى الطرفية Terminal ثم قم بالضغط على كلا من الزر Ctrl مع الحرف r ستظهر لك عبارة بالشكل التالى :

كود :PHP

```
 (reverse-i-search)`':
```

بعد ذلك قم بادخل مثلا الأمر ls ماذا تلاحظ ؟

بمجرد أن تقوم بإدخال حرفى ّ الأمر ستظهر على الفور جملة اول أمر تم كتابته قم ذلك وللبحث عن مزيد من جمل الأوامر التى تحتوى الأمر ls قم بالضغط فى نقس اللحظة على Ctrl+r عدة مرات لتحصل على النتيجة التى تريدها .

ملحوظة : لاحظ أن الخاصية Ctrl+r تقوم بالبحث عن الأوامر التى توجد داخل الملف bash_history. فقط .

الأمر alias

هل سئمت كثيرا من كتابة بعض الأوامر مع بعض الخيارات التى تأتى معها بشكل مستمر ؟ إذا هنا يأتى دور الأمر alias فترجمة الكلمة تعنى الإسم المستعار ، فهو بشكل ما يدل على الإسم الأصلى له فمثلا

باستمرار أنت تستخدم الأمر ls مع الخيار l- لعرض التصاريح الموجودة على الملفات لكن فى بعض الأحيان يتملكك الكسل وتريد تسهيل المهمة لكن لا تعرف كيف ، لذا دور الأمر alias أن يسهل لك الصعاب ويوفر عليك بعض الوقت عند كتابة الأوامر ولتتضح الأمور نبدأ فى تطبيق بعض الأمثلة العملية :

فى البداية سنقوم باستعراض كل ال aliases الموجودة على التوزيعة الخاصة بك وهل توجد أم لا وذلك بتنفيذ الأمر التالى من محاكى الطرفية :

كود PHP:

```
muhammad@ubuntu:~$ alias
alias ls='ls --color=auto'
```

كما ترى يوجد لدى alias واحد فقط يخص الأمر ls ويأتى مع الخيار color=auto-- ، وقد يختلف الأمر معك وتظهر لديك عدة aliases ، ويتم الإحتفاظ بتلك ال aliases إما داخل الملف bashrc. أو داخل ملف مستقل اسمه bash_aliases. ، لكن فى العادة يتم الإحتفاظ باى alias تقوم بإنشائه داخل الملف bashrc. والذى يكون مسئول عن ضبط خيارات الطرفية للمستخدم الذى يملك ذلك الملف

سأقوم بتحرير باقى تلك المشاركة قريبا